MASTER EN SCIENCES MATHÉMATIQUES

Application des préconditionneurs à mémoire limitée au recalage d'images médicales

Parmentier, Sophie

Award date:
2017

Awarding institution:
Universite de Namur

UNIVERSITE DE NAMUR

Faculté des Sciences

APPLICATION DES PRECONDITIONNEURS A MEMOIRE LIMITEE AU RECALAGE D'IMAGES MEDICALES

Promoteur : Annick SARTENAER

Mémoire présenté pour l'obtention

du grade académique de master en Sciences mathématiques à finalité spécialisée

Sophie PARMENTIER

Juin 2017

Remerciements

Je souhaite adresser mes remerciements les plus sincères aux personnes qui m'ont apporté leur aide et qui ont contribué, de près ou de loin, à l'élaboration de ce mémoire.

Je tiens à remercier tout particulièrement Annick Sartenaer qui, en tant que promotrice de ce mémoire, s'est toujours montrée à l'écoute tout au long de la réalisation de ce travail. Merci à elle pour le temps qu'elle m'a consacré et la confiance qu'elle m'a accordée.

Je remercie également Hubert Meurisse, radio-physicien et ingénieur biomédical au Service Nucléaire du Centre Hospitalier Universitaire de Mont-Godinne de m'avoir permis de réaliser ce travail en proposant ce sujet.

Je remercie également de tout cœur Justin Buhendwa Nyenyezi et Jean Tshimanga pour leurs conseils qui m'ont toujours permis de trouver une solution à chacun des problèmes rencontrés.

J'adresse également mes plus sincères remerciements à toutes les personnes qui, ces cinq dernières années, m'ont fait voir le bon côté des choses quand il le fallait. Plus particulièrement à mes parents, merci de m'avoir donné tant d'opportunités et de m'avoir si bien encouragée. Pour terminer, un grand merci à mon soutien de taille, ma soeur, Steph, pour toutes ces heures de pause et de rigolade amplement nécessaires.

Résumé

Dans le domaine médical, les professionnels sont amenés à combiner de l'information contenue dans des images obtenues par différentes méthodes d'acquisition et différentes modalités. Le processus de recalage d'images permet l'alignement des images et leur superposition au sein d'un même domaine. La distance entre les images à recaler est quantifiée par un critère de ressemblance qui doit être le plus petit possible. Ainsi, le recalage d'images s'apparente à un problème d'optimisation dit problème aux moindres carrés non linéaires résolu au moyen de la méthode de Gauss-Newton. Le but de ce mémoire est, dans un premier temps, d'introduire la classe des préconditionneurs à mémoire limitée pour ensuite détailler trois instances qui permettront d'accélérer la résolution. Dans un second temps, il étudie l'implémentation de ces instances dans la librairie FAIR fournissant des algorithmes de recalage d'images. Enfin, une stratégie de résolution et une application de recalage sont proposées.

Mots-clés : imagerie médicale, recalage d'images, librairie FAIR, préconditionneurs à mémoire limitée, Gauss-Newton, optimisation.

Abstract

In the medical field, professionals are brought to combine information contained in images obtained by different acquisition methods and different modalities. The image registration process allows the alignment of images and their superimposition within the same domain. The distance between the images is quantified by a criterion of similarity which must be as small as possible. The image registration is similar to an optimization problem known as a nonlinear least squares problem solved using the Gauss-Newton method. The aim of this master's thesis is first introduce the class of Limited Memory Preconditioners and then to detail three instances that will improve the velocity of the resolution. Further, it studies the implementation of these instances in the FAIR library providing image registration algorithms. Finally, a resolution strategy and an application of image registration are proposed.

Keywords : medical imaging, image registration, FAIR library, Limited Memory Preconditioners, Gauss-Newton, optimization.

Table des matières

Introduction

Ce mémoire s'inscrit dans la continuité d'une collaboration avec le Centre Hospitalier Universitaire de Mont-Godinne et plus particulièrement avec le service de médecine nucléaire. L'imagerie médicale est un domaine très important de la médecine moderne. Elle intervient dans la prise de décision, la planification de traitement mais aussi le suivi thérapeutique du patient. Les professionnels de la santé ont très souvent recours à l'analyse de plusieurs images obtenues à différents moments ou selon différentes techniques d'acquisition. Le processus de recalage permet de placer deux images dans un référentiel commun pour ainsi combiner et/ou comparer l'information respective des images. Si l'une d'elles est considérée comme image de référence, ce processus requiert l'estimation d'une transformation spatiale qui déforme au mieux l'autre image afin qu'elle lui soit le plus similaire possible. Deux méthodes de recalage existent, il s'agit des méthodes rigide et non rigide. La première nécessite uniquement des transformations du type rotation et translation. Quant à la seconde, elle permet de recaler en déformant l'image, par exemple, à l'aide de transformations élastiques. Un critère de similarité est utilisé pour quantifier la ressemblance entre les deux images. La qualité de la transformation dépend des méthodes d'acquisition mais aussi de la zone du corps humain concernée. L'objectif est de trouver la transformation qui minimise la distance entre les deux images. L'estimation de la transformation spatiale revient dès lors à résoudre un problème d'optimisation.

Ce mémoire est articulé en quatre chapitres. Le premier chapitre introduit l'imagerie médicale ainsi que différentes techniques de recalage. Nous expliquerons les notions importantes à l'aide d'exemples illustratifs concrets dans le cadre de recalage en deux dimensions. Ensuite, nous poursuivrons avec le fondement du processus de recalage et la formulation du problème de recalage comme un problème d'optimisation. Celui-ci est en général de taille importante et nécessite des méthodes de résolution itératives. Le second chapitre présente différents outils d'optimisation qui nous seront utiles dans la suite pour accélérer la résolution. En effet, le problème du recalage s'apparente au problème des moindres carrés non linéaires qui peut être résolu à l'aide de la méthode de Gauss-Newton. Cette méthode implique la résolution de systèmes linéaires symétriques définis positifs par des méthodes de type gradient conjugué. Ces systèmes sont de grande taille et sont sujets à un mauvais conditionnement ralentissant la résolution du problème. L'idée est de construire un préconditionneur approchant l'inverse de la matrice du système et améliorant le conditionnement. Dans ce contexte, le troisième chapitre introduit trois instances de la classe des préconditionneurs à mémoire limitée. Nous étudierons leurs propriétés afin de les utiliser au mieux comme préconditionneur. Finalement, le dernier chapitre considère l'application de ces préconditionneurs à une résolution concrète. Nous utiliserons la librairie FAIR [1] et les algorithmes de recalage d'images qu'elle propose. Nous implémenterons les trois instances décrites dans le chapitre précédent et proposerons une stratégie de résolution.

Chapitre 1

Le recalage d'images médicales

L'imagerie médicale est un domaine important de la médecine puisqu'elle offre la possibilité d'explorer le corps humain en 2 ou 3 dimensions que ce soit à des fins diagnostics ou thérapeutiques [2]. La problématique que nous aborderons concerne l'analyse et le traitement de plusieurs images dans un référentiel commun. Ce problème est connu sous le nom de recalage et requiert l'estimation d'une transformation géométrique permettant la superposition spatiale d'attributs similaires correspondant aux images [3].

Ce chapitre introduit les grandes lignes de l'imagerie médicale pour ensuite s'intéresser au recalage d'images médicales. Dans ce contexte, il est intéressant de pouvoir combiner et comparer les différentes informations obtenues par des méthodes d'acquisition, qu'elles soient identiques ou différentes. Ainsi, nous aborderons des exemples concrets de recalage d'images médicales pour illustrer différentes notions.

Nous aborderons les fondements théoriques relatifs aux approches géométrique et iconique pour guider le recalage. Une formulation du problème de recalage sera proposée pour finalement être adaptée à une stratégie d'optimisation spécifique. Nous terminerons par l'introduction du rôle de l'interpolation et son enjeu dans le processus.

1.1 L'imagerie médicale

L'imagerie médicale a pour objectif de créer une représentation visuelle d'une information à caractère médical [4]. Elle doit être suffisamment précise de manière à apporter un diagnostic le plus assuré possible. En d'autres termes, l'objectif est de concentrer une grande quantité d'informations issues d'un grand nombre de mesures obtenues selon un ou plusieurs types d'acquisition. Ces images sont acquises selon 3 types de plans différents qui permettent de désigner l'angle sous lequel le corps est observé.

1.1.1 Les systèmes de référence en anatomie

La *position anatomique* de référence est définie par celle du corps humain vivant en position debout, les pieds réunis, les paumes tournées vers le sol comme illustré sur la figure 1.1. A partir de cette position, trois groupes principaux de plans permettent de définir un *système de référence*. Ces groupes sont les suivants :

- Le *plan axial* ou *plan transverse* : tout plan perpendiculaire à l'axe vertical du corps ou des membres.
- Le *plan sagittal* : tout plan vertical divisant d'avant en arrière le corps en deux parties : droite et gauche. Le plan sagittal passant par le centre du corps et le divisant en deux parties égales (droite et gauche) est appelé plan sagittal médian.

- Le *plan coronal* ou *plan frontal* : tout plan vertical parallèle à la face ventrale du corps et le divisant en deux parties : ventrale et dorsale.

Ces définitions sont tirées de la librairie médicale [5]. La description ou la représentation anatomique d'une zone corporelle doit toujours comporter son orientation par rapport aux trois plans de référence : horizontal, médio-sagittal et frontal. Notons qu'il existe deux conventions, anatomique et radiologique, qui diffèrent dans la représentation de la coupe transversale. La première impose qu'elle soit représentée en vue supérieure : la gauche de l'observateur correspond à la gauche du patient. A l'inverse, la seconde impose une représentation en vue inférieure : la gauche de l'observateur correspond à la droite du patient.

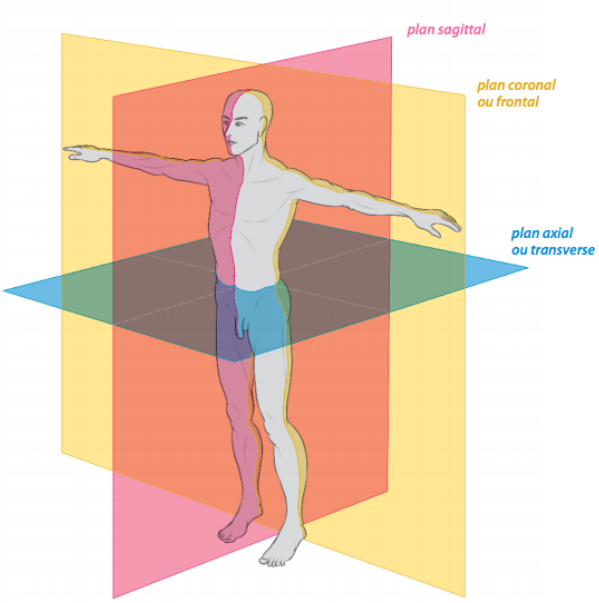


FIGURE 1.1 – Plan anatomique. [5]

1.1.2 Les techniques d'acquisition

L'imagerie médicale fait usage de nombreuses techniques d'acquisition permettant d'obtenir des informations lors d'un examen. Nous pouvons dégager deux grands types d'imagerie :

- L'imagerie *structurelle* qui concerne l'anatomie des organes et permet d'obtenir des informations comme leur taille et leur localisation.
- L'imagerie *fonctionnelle* qui s'intéresse, par exemple, au métabolisme, à la physiologie d'un organe.

Ces deux techniques d'imagerie sont illustrées à la figure 1.2 avec des images du cerveau.

En médecine nucléaire, les méthodes d'imagerie structurelle couramment employées sont la résonance magnétique nucléaire, les ultra-sons dans le cadre d'échographies et les rayons X. Ces derniers sont employés en radiologie conventionnelle et dans les CT-scans pour ne citer qu'eux. Parmi les méthodes d'imagerie fonctionnelle, nous pouvons citer des techniques basées sur l'émission de rayons gamma. Pour effectuer les examens relatifs à cette méthode, des traceurs radioactifs sont administrés par injection et vont se concentrer dans des régions d'intense activité métabolique, caractéristique d'une tumeur par exemple. Citons aussi l'IRM dite fonctionnelle et d'autres techniques électro-physiologiques.

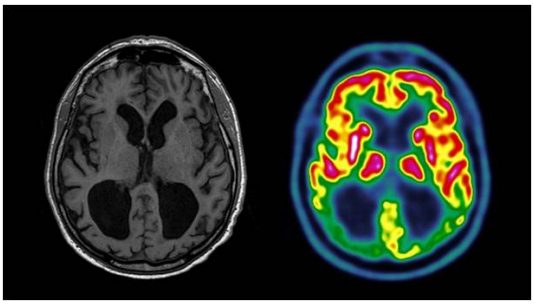


FIGURE 1.2 – Imagerie structurelle (à gauche) et imagerie fonctionnelle (à droite) du cerveau. [6]

Dans la suite de ce mémoire, une technique d'acquisition sera aussi nommée *mode*. Ces techniques ne fournissent pas les mêmes informations sur les organes d'un patient, elles sont complémentaires. Le médecin peut être amené à comparer des aspects structurel et fonctionnel du corps humain pour poser son diagnostic. Afin de profiter des particularités de plusieurs modes, il faut être capable de combiner l'information de chaque image. La fusion d'images, illustrée à la figure 1.3, implique implicitement que les images soient dans un même repère géométrique. Elles sont alors spatialement comparables.

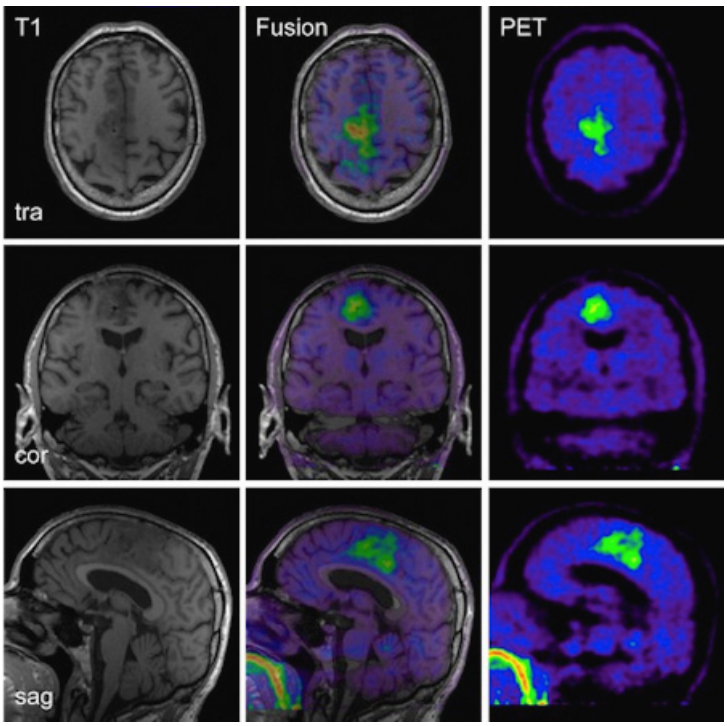


FIGURE 1.3 – Exemple d'imagerie multimodale. [7]

L'exemple de la figure 1.3 représente des images PET/IRM d'un patient de 60 ans avec une tumeur astrocytaire très peu différenciée et une extension de la tumeur à droite. Par ligne, nous avons les coupes transversales, coronales et sagittales. Pour ce qui est des colonnes, la première est composée de données IRM et la troisième de données PET-scan. La colonne du milieu représente la fusion des deux autres colonnes, c'est-à-dire l'information regroupée de celles-ci. Notons que ces données ont été acquises simultanément.

1.2 Le recalage

Prenons la définition de recalage selon Wikipédia [8]. En traitement d'images, le recalage est une technique qui consiste en la mise en correspondance d'images, ceci afin de pouvoir comparer ou combiner leurs informations respectives. Cette mise en correspondance se fait par la recherche d'une transformation géométrique permettant de passer d'une image, dite "mouvante", à l'autre, dite "fixe".

Comme nous l'avons vu dans la section précédente, plusieurs images acquises selon des modalités différentes sont parfois utilisées pour effectuer un seul diagnostic. Cela requiert qu'elles soient dans un même repère. Or, entre les acquisitions, le patient peut avoir bougé. Il se peut aussi qu'un battement de cœur ou une respiration profonde impacte les données. L'intérêt du recalage est d'aligner les acquisitions dans un même repère.

Supposons que nous disposons de deux images à recaler. Si l'on considère l'une d'elle comme image de référence, ce processus consiste à déterminer la meilleure transformation spatiale qui déforme au mieux l'image restante pour qu'elle lui soit le plus similaire possible. Un exemple de recalage est donné à la figure 1.4. Les informations recueillies doivent être fusionnées dans un même repère pour permettre au praticien d'effectuer des analyses et comparaisons nécessaires à l'établissement de son diagnostic.

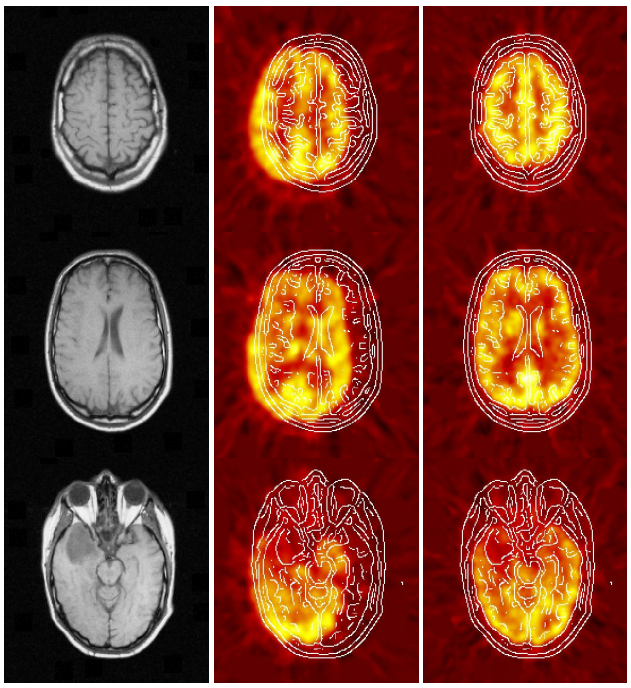


FIGURE 1.4 – Exemple de recalage avec 3 plans de coupe différents de gauche à droite : la coupe IRM, la coupe correspondante obtenue par un PET-scan non recalée et la même coupe recalée. La coupe acquise par le PET-scan est superposée avec les contours extraits de l'IRM afin de visualiser la qualité du recalage. [9]

Deux contextes peuvent être distingués, il s'agit des recalages intrasujet et intersujet. Dans le premier cas, les images proviennent d'un même individu. Quant au second, il met en correspondance des images provenant de sujets différents ou d'un sujet et d'un modèle de référence appelé *atlas*.

1.2.1 Le recalage multimodal

Un exemple de recalage multimodale a été proposé à la figure 1.3. Nous nous trouvons dans le cadre d'images intrasujet car il s'agit du cerveau d'un même patient sur chaque cliché. L'information structurelle est fournie par l'IRM et l'information fonctionnelle par le PET-scan. La combinaison des deux va permettre d'associer une activité cérébrale à une structure. En effet, l'anatomie du cerveau du patient est donnée par l'IRM alors que le PET-scan permet d'identifier la présence d'une tumeur (en vert dans le cerveau). Le recalage n'est autre que la combinaison des deux images. Elle permet de connaître la localisation de la tumeur en utilisant les caractéristiques fonctionnelles du PET-scan et les caractéristiques structurelles de l'IRM.

Le recalage multimodal utilisant des images de sujets différents est très peu utilisé en médecine nucléaire. Nous n'en parlerons pas dans le cadre de ce mémoire.

1.2.2 Le recalage monomodal

Il est parfois nécessaire de recaler des images acquises par un même mode. Dans ce cas, il peut s'agir d'un recalage pour des images issues d'un même patient. Nous parlerons alors de recalage monomodal et intrasujet.

(a) (b) (c)

FIGURE 1.5 – Exemple monomodal - intrasujet. [10]

En pratique, le recalage d'images d'un cerveau adulte après un infactus permet d'analyser l'évolution et l'étendue de la zone atrophiée. Un autre exemple est donné à la figure 1.5 qui illustre l'évolution d'une tumeur connue sous le nom de méningiome. Les images (a) et (b) sont placées dans un même repère et représentent des coupes IRM avec 13 mois d'écart. La différence des deux images est représentée à l'image (c) ce qui permet de conclure médicalement que la maladie a progressé et que l'exophtalmie s'est amplifiée.

Les images médicales peuvent varier très fortement entre des individus car la taille et la forme de leurs organes diffèrent. Il est important de pouvoir quantifier ces variations. Une des approches les plus connues est celle de l'atlas qui réfère à un modèle spécifique pour une certaine population. La création et l'utilisation d'un atlas utilise le recalage monomodal avec des images provenant de patients différents. Nous parlerons de recalage monomodal mais cette fois intersujet.

L'idée est de recaler des données cliniques afin qu'elles puissent être comparées à un atlas. De gauche à droite sur la figure 1.6, nous avons l'IRM du cerveau d'un patient puis le cerveau référencé dans l'atlas. Finalement, nous avons le recalage du cerveau du patient qui peut être comparé à l'atlas.

FIGURE 1.6 – Exemple de recalage dans le cadre d'un atlas. [11]

1.3 Fondements théoriques

Le problème de recalage d'images est facile à poser. Nous allons introduire un exemple simple qui nous permettra de fixer des notions importantes pour la suite de ce mémoire. Deux approches permettant de traiter des attributs de l'image seront développées.

1.3.1 Introduction : recalage rigide

La figure 1.7 est composée de deux images M et N d'une même scène, acquises par un même capteur. Ces acquisitions sont espacées d'un cours laps de temps, nous supposerons que la position du capteur a légèrement changé. Le recalage va consister à trouver la transformation géométrique **T** qui met en correspondance les coordonnées **x** aux coordonnées **y** appartenant respectivement aux espaces V_M et V_N telle que :

$$\mathbf{x} = \mathbf{T}(\mathbf{y}).$$

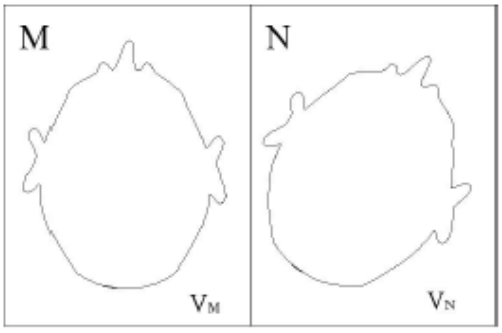


FIGURE 1.7 – Illustration simplifiée du problème de recalage. [12]

Nous ne nous intéressons pour le moment qu'à la transformation rigide qui est largement utilisée en recalage d'images médicales et plus particulièrement pour les images cérébrales. Une transformation rigide préserve les angles et ne déforme pas l'image. Celle-ci peut s'écrire $\mathbf{T}(x, y, z)$ et est composée d'une translation suivant x, y, ou z et d'une rotation suivant x, y, ou z [13].

Nous pouvons identifier deux éléments importants dans le recalage d'images qui dépendent de l'application médicale et de la méthode d'acquisition. Pour l'instant, nous signalons simplement qu'il s'agit de l'espace et la nature de la transformation géométrique.

Il existe plusieurs méthodes de recalage qui peuvent être classées selon les *attributs* de l'image aussi appelés primitives. L'estimation de la transformation se fait sur base de plusieurs types d'informations. Par exemple, les primitives géométriques et l'intensité des images, qui donneront lieu respectivement aux approches géométrique et iconique.

1.3.2 Approche géométrique

Cette méthode intuitive fut dans les premières à être proposées. Elle se base sur l'extraction de caractéristiques géométriques des deux images qui sont, par exemple, des points, des surfaces ou des volumes. Le recalage mettra en relation la position d'une caractéristique de l'image mouvante avec la position de la même caractéristique de l'image fixe. Dans la suite, nous considérerons que les caractéristiques sont des points.

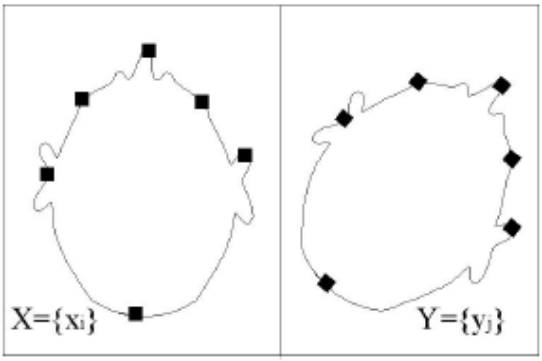


FIGURE 1.8 – Extraction de points homologues dans le cadre de l'approche géométrique. [12]

La transformation géométrique recherchée doit minimiser l'écart entre les deux images, ce qui signifie que la distance entre chaque couple de points doit être minimale au sens des moindres carrés. La notion d'optimisation au sens des moindres carrés sera abordée plus loin. La transformation est de la forme :

$$\mathbf{T} : \mathrm{V}_M \rightarrow \mathrm{V}_N : \mathbf{y} \rightsquigarrow \mathbf{T}(\mathbf{y})$$

où V_M et V_N sont respectivement les domaines de l'image mouvante et de l'image fixe. La position $\mathbf{x}$ est une position dans le domaine de l'image mouvante.

Il est évident que le nombre de paires de points de correspondance utilisés influence fortement la qualité du recalage. Plus ils sont nombreux, plus la transformation géométrique sera fine et donc le recalage sera meilleur.

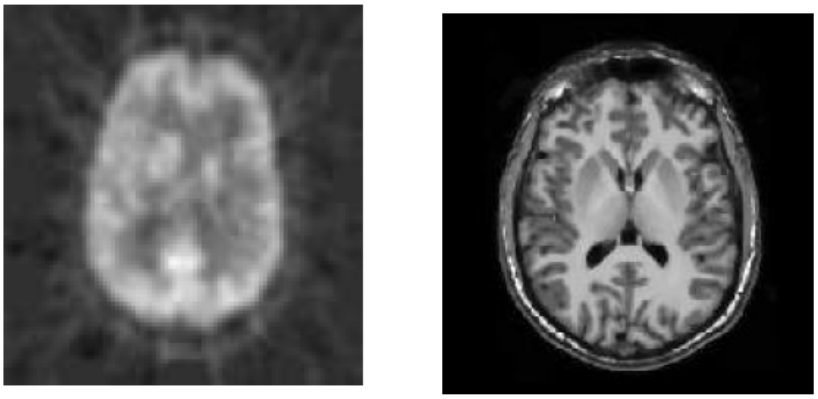
Cette approche nécessite moins de calculs, les primitives utilisées portent une information de haut niveau et les différences d'intensité entre deux modes ne posent pas problème. Cependant, la difficulté de cette approche est l'identification des paires de points homologues dont l'extraction n'est pas toujours précise. En imagerie médicale, l'expertise du radiologue est requise pour extraire les primitives géométriques. Ce processus de recherche de points d'intérêt est manuel et fastidieux. De ce fait, plusieurs études tentent de rendre cette démarche automatique en s'affranchissant de l'expertise manuelle.

1.3.3 Approche iconique

L'approche iconique est fondée sur la comparaison iconique des images. Les primitives ne sont plus géométriques mais correspondent à des vecteurs 3D en 2 dimensions et toute l'information portée par l'image est utilisée. Pour un point déterminé, un vecteur contient la position du point et son intensité. Il n'y a plus d'étape préalable de la compréhension de la géométrie de l'objet, ce qui qualifie la méthode de méthode de bas niveau. Il faut, par contre, comprendre le lien d'intensité entre deux images c'est-à-dire entre les grandeurs physiques mesurées par leur mode respectif.

Cette approche est plus adaptée au recalage multimodal, contrairement à l'approche géométrique qui, elle, s'adapte mieux au recalage monomodal. D'après P. Cachier [14], la transformation recherchée a deux objectifs de nature différente. D'une part, un objectif iconique : cette transformation doit recaler les images afin que leurs intensités s'expliquent mutuellement le mieux possible. D'autre part, un objectif géométrique : la transformation estimée doit être lisse, afin de contraindre l'appariement, et de traduire, dans une certaine mesure, la réalité physique de la continuité du milieu.

Le succès de l'approche iconique réside dans les avancées considérables qu'elle a permis dans le domaine du recalage multimodal, lorsque l'approche géométrique rencontrait des difficultés majeures. Prenons l'exemple de la figure 1.9 qui a pour objectif le recalage IRM/TEP d'un cerveau. D'un côté l'IRM fournit une image anatomique tandis que la tomographie par émission de positrons (TEP) fournit une image fonctionnelle [14]. La difficulté rencontrée lors de l'utilisation de l'approche géométrique provient de l'image acquise par TEP. Les informations morphologiques sont difficilement identifiables en raison de la faible résolution. Il est donc difficile d'identifier des points de repère qui serviront à trouver la transformation. Ainsi, l'approche iconique permet de recaler en utilisant la relation entre les intensités des deux images.



(a) Acquisition TEP. (b) Acquisition IRM.

FIGURE 1.9 – Problème de recalage multimodal intrapatient complexe. [14]

1.4 Formulation du problème de recalage

Cette section a pour but de formuler le problème de recalage comme un problème d'optimisation. Il s'agit de donner au lecteur une vue d'ensemble de la problématique, en explicitant le problème de façon continue, pour ensuite y inclure le terme de régularisation. Finalement, une fois la stratégie *discrétiser puis optimiser* proposée, nous donnerons la fonction objectif qui guidera l'application dans la suite du mémoire.

1.4.1 Formalisme continu

Nous allons définir de manière continue le recalage de deux images : $\mathcal{R}$, l'image fixe et $\mathcal{T}$, l'image mouvante. Le recalage est un processus qui permet de déterminer la transformation spatiale la plus adéquate, afin qu'après transformation, l'une d'elle soit le plus possible similaire à l'autre. Une image peut se définir de la façon suivante.

Définition 1.1. Image [15]
Une image est une correspondance continue entre un domaine Ω *et des nombres réels telle que*

$$i : \Omega \subset \mathbb{R}^d \to \mathbb{R}$$

et dont la dimension est d.

En imagerie médicale, la dimension d est 2 ou 3. Cette définition nous permet d'écrire

$$\mathcal{R} : \Omega_{\mathcal{R}} \subset \mathbb{R}^d \to \mathbb{R} \quad \text{et} \quad \mathcal{T} : \Omega_{\mathcal{T}} \subset \mathbb{R}^d \to \mathbb{R}.$$

Dans la section 1.3, nous avions introduit le recalage rigide dont la transformation géométrique n'utilise que des rotations et/ou translations. Le recalage non-rigide sera au centre de la suite de ce mémoire. Il concerne les recalages d'images pour lesquelles toutes sortes de transformations élastiques et déformables sont envisageables. On parle alors de *modèle de déformation*.

La transformation non rigide recherchée peut s'écrire

$$y : \Omega_{\mathcal{R}} \to \Omega_{\mathcal{T}} : s \rightsquigarrow y(s) = u(s) + s$$

avec u, une déformation. La transformation associe à chaque point s de l'image fixe, les coordonnées $y(s) = u(s) + s$ dans l'image mouvante. Cette transformation doit appartenir à un espace de transformations admissibles noté Y.

Maintenant que nous avons défini certains éléments à utiliser pour guider le recalage, la ressemblance entre les deux images doit pouvoir être calculée. Ainsi, un critère de similarité va mesurer la distance entre les deux images et quantifier la ressemblance. Ce critère devrait théoriquement être minimal lorsqu'il existe une parfaite correspondance entre l'image mouvante et l'image fixe. Dans le cadre de ce mémoire, nous utiliserons comme mesure de dissimilarité la somme des carrés des différences des intensités. Le terme *dissimilarité* et non pas *similarité* est utilisé car nous cherchons à minimiser le critère.

La phase principale du recalage consiste à trouver la transformation optimale $\hat{y}$ qui minimise le critère de dissimilarité sur l'espace des transformations admissibles Y. Le problème d'optimisation se formule de la manière suivante

$$\hat{y} = \underset{y \in Y}{\arg\min} \ \mathcal{D}[\mathcal{T}(y), \mathcal{R}] \tag{1.1}$$

avec la fonction objectif

$$\mathcal{D}[\mathcal{T}(y), \mathcal{R}] = \frac{1}{2} \int_{\Omega_{\mathcal{R}}} \big(\mathcal{T}(y(x)) - \mathcal{R}(x)\big)^2 \, \mathrm{d}x. \tag{1.2}$$

La figure 1.10 est inspirée de [3] et décrit simplement le processus de recalage que nous venons de présenter. Une transformation y est calculée pour ensuite être appliquée à $\mathcal{T}$. La ressemblance entre les deux images est quantifiée par la mesure de dissimilarité. Selon le résultat obtenu, la transformation est mise à jour afin de rendre l'image mouvante de plus en plus similaire à l'image fixe ce qui correspond à la phase d'optimisation.

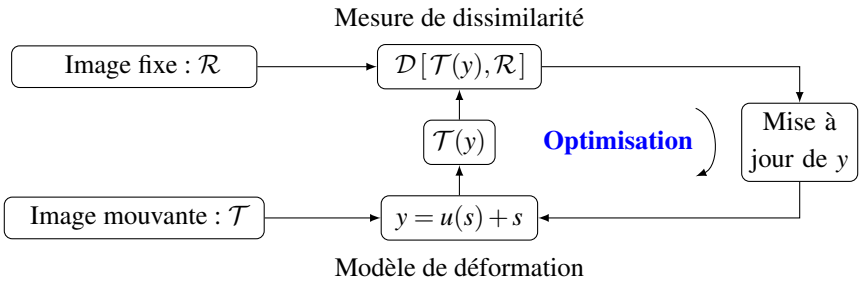


FIGURE 1.10 – Processus de recalage.

1.4.2 Terme de régularisation

La transformation recherchée lors du processus de recalage non rigide dépend souvent de nombreux paramètres. Dès lors, le nombre de degrés de liberté s'avère souvent important et l'information portée par l'image peut s'avérer insuffisante pour estimer correctement l'ensemble des paramètres. Le problème d'optimisation (1.1) est alors dit *mal posé au sens d'Hadamard*. En d'autres termes, s'il existe une solution, alors cette solution est unique et dépend continuellement des données, sinon le problème est mal posé [15]. L'ajout de contraintes supplémentaires sur la transformation va permettre de garantir l'unicité de la solution. De plus, cela lisse le modèle de déformation dans les zones homogènes, pour lesquelles l'information ne permet pas d'estimer correctement les paramètres permettant d'obtenir la transformation.

Le terme de régularisation

$$\mathcal{S}[y] = \frac{1}{2}\int_{\Omega_R} \left(\mathcal{B}(y - y^{ref})\right)^2 \mathrm{d}x \tag{1.3}$$

intervient dans le choix de la déformation. Il va pénaliser les comportements irréguliers du modèle de déformation et ainsi éviter l'estimation de petites déformations parasites qui ne réduisent la fonction (1.2) que d'une petite quantité. Ce terme est basé sur la norme L_2 de la dérivée du déplacement $y - y^{ref}$ obtenue via l'opérateur différentiel $\mathcal{B}$. Le choix de l'opérateur nécessite une connaissance à priori de la forme de la transformation. Différentes approches existent, il s'agit des approches élastiques, fluides, splines et par déformations libres qui sont largement détaillées dans [16]. Quant au terme y^{ref}, celui-ci permet un biais depuis une solution particulière. Par exemple, cette solution peut être le résultat d'un recalage rigide effectué avant un recalage non rigide.

Dans le contexte du recalage d'images médicales, il est difficile de motiver le choix d'un terme de régularisation plutôt qu'un autre. Dans la littérature, les termes de régularisation sont liés à des considérations mécaniques qui n'ont en général aucune justification biologique.

Finalement, la fonction objectif globale du recalage peut s'écrire

$$\begin{aligned} \mathcal{J}[y] &= \mathcal{D}[\mathcal{T}(y),\mathcal{R}] \;+\; \mathcal{S}[y] \\ &= \frac{1}{2}\int_{\Omega_R} \left(\mathcal{T}(y(x)) - \mathcal{R}(x)\right)^2 \mathrm{d}x \;+\; \frac{1}{2}\int_{\Omega_R} \left(\mathcal{B}(y - y^{ref})\right)^2 \mathrm{d}x \end{aligned} \tag{1.4}$$

et résulte de l'association de deux termes : le critère de dissimilarité entre les deux images et la régularisation de la déformation. Le problème d'optimisation (1.1) devient

$$\hat{y} = \underset{y \in Y}{\arg\min}\ \mathcal{J}[y]. \tag{1.5}$$

1.4.3 Stratégie d'optimisation

Le problème de recalage que nous venons de présenter est formulé de façon continue. Or, en pratique, les images sont recalées sur base de leur représentation discrète. La stratégie d'optimisation que nous utiliserons consiste à *discrétiser puis optimiser*. Concrètement, le problème continu est remplacé par un problème discret tout en tenant compte de la régularisation pour ensuite appliquer des méthodes d'optimisation afin d'obtenir une solution suffisamment proche de celle du problème continu.

La figure 1.11 illustre la numérisation de l'IRM d'un cerveau sur l'image (a) dont une zone est encadrée. Un agrandissement de celle-ci est proposée sur l'image (b) avec, par-dessus, deux types de grilles. L'image (c) représente la numérisation de l'IRM en (a) selon la grille nodale. L'image (d) représente la numérisation mais cette fois selon la grille centrée. Cet exemple montre la différence flagrante que donne le résultat des deux modèles de numérisation. Notons qu'il existe un troisième type de grille dite échelonnée qui n'est pas représentée ici.

La résolution du problème du recalage avec cette approche nécessite de discrétiser la fonction objectif (1.4). Ainsi, nous obtenons la fonction objectif discrète suivante

$$J[y] = \frac{1}{2}\,\|\,\mathcal{T}(y) - \mathcal{R}\,\|_2^2 \;+\; \frac{1}{2}\,\|\,\mathcal{B}(y - y^{ref})\,\|_2^2. \tag{1.6}$$

Résoudre le problème de recalage discrétisé revient à trouver la transformation y qui minimise (1.6). Le problème peut ainsi être formulé de la manière suivante

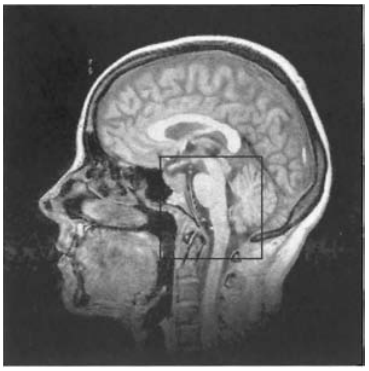
$$\hat{y} = \underset{y \in Y}{\arg\min}\ J[y]. \tag{1.7}$$

Le problème (1.7) se résout numériquement comme un problème des moindres carrés non linéaires.

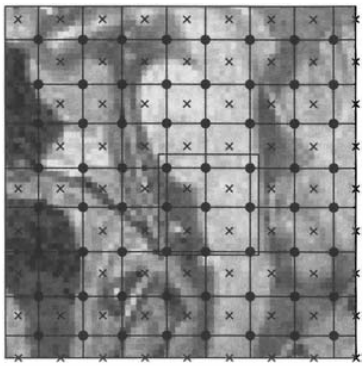
1.5 Interpolation et ré-échantillonnage

Jusqu'à présent, un élément clé du processus de recalage n'a pas été mentionné, il s'agit de l'interpolation. Son rôle principal est de permettre de passer d'une image discrète à une image continue. Dans le cadre de l'approche iconique, après application de la transformation à l'image mouvante $\mathcal{T}$, les valeurs d'intensité doivent être calculées pour les différents points obtenus afin de quantifier la ressemblance.

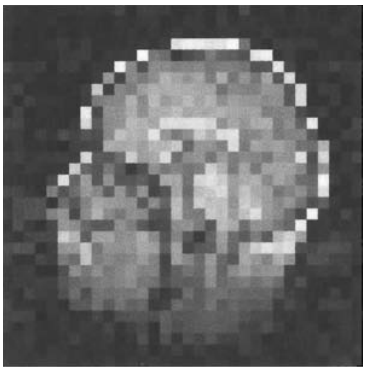
Dans un premier temps, nous pouvons envisager l'utilisation d'une grille associée à l'image mouvante à recaler. L'approche utilisant une transformation directe consiste à définir pour l'image mouvante une grille similaire à celle de l'image fixe. Une fois la transformation appliquée, nous disposons d'une image déformée dont les points de la grille ne correspondent pas nécessairement avec les points de la grille de l'image de référence. Il faut donc pallier au problème de l'évaluation des valeurs d'une image pour des coordonnées non entières en interpolant.



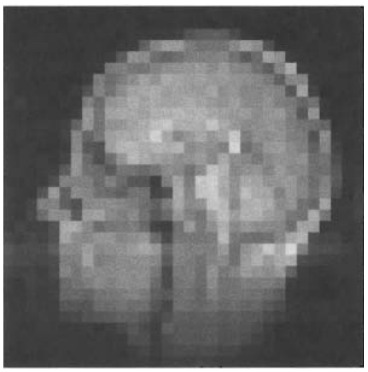
(a) IRM avec identification d'une zone.



(b) Détails de la zone encadrée en (a) avec ● pour la grille nodale et × pour la grille centrée.



(c) Numérisation à partir de la grille nodale.



(d) Numérisation à partir de la grille centrée.

FIGURE 1.11 – Exemple de numérisation (32×32 pixels). [16]

En pratique, nous utilisons la transformation inverse, qui associe à chaque point de la grille de l'image déformée, les coordonnées du point associé sur la grille de l'image fixe. Dans le cas discret, il est souhaitable que les deux images aient la même grille d'échantillonnage.

Les deux approches sont représentées sur les figures 1.12 et 1.13. Il est opportun de travailler avec la transformation inverse, car l'interpolation se fait sur une grille régulière. L'interpolation sera donc plus efficace en termes de coût calculatoire et de qualité. En imagerie, ce concept s'appelle le *ré-échantillonnage*. Il est utilisé lorsque le déplacement des points n'est pas proportionnel à la grille. S. Oze [17] donne plus de détails quant au fonctionnement de ce concept.

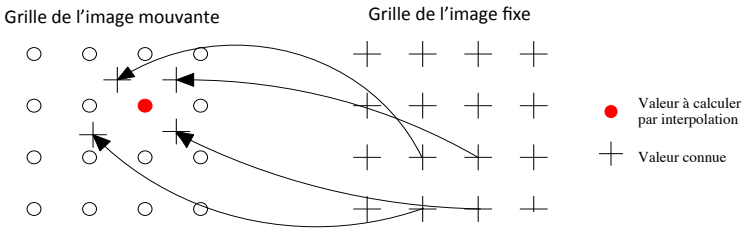


FIGURE 1.12 – Transformation directe. [3, Modifiée]

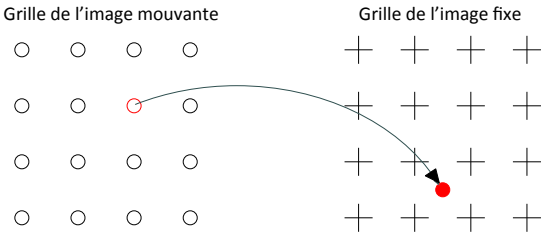


FIGURE 1.13 – Transformation inverse. [3, Modifiée]

1.6 Conclusion

Ce chapitre vient de traiter le problème complexe qu'est le recalage d'images médicales. Il a commencé par présenter l'imagerie médicale, pour ensuite s'intéresser à la problématique du recalage dans ce contexte. Deux grandes approches ont été abordées, il s'agit des approches géométrique et iconique. L'application proposée dans la suite de ce mémoire concerne la seconde approche.

Par la suite, une présentation des enjeux du recalage d'images a été donnée. Le recalage a été formulé dans le cadre non-rigide. Ainsi, la fonction objectif proposée pour poser le problème d'optimisation intègre un terme dit de régularisation en plus du terme de mesure de dissimilarité. Celui-ci permet d'introduire des hypothèses quant à la régularité du modèle de déformation. Nous avons présenté une stratégie qui discrétise puis optimise pour estimer le modèle de déformation. Des méthodes d'optimisation doivent maintenant être introduites. En effet, le problème du recalage nécessite un algorithme de résolution des moindres carrés non linéaires afin d'obtenir la transformation optimale. A chaque boucle de cet algorithme, nous allons devoir résoudre itérativement un système linéaire symétrique défini positif. Le chapitre suivant traite des différentes techniques d'optimisation que nous utiliserons pour résoudre ces systèmes.

Chapitre 2

Optimisation

L'objectif de ce chapitre est d'introduire des outils d'optimisation. Il est courant de devoir optimiser des problèmes de grande taille. Ceux-ci induisent des matrices creuses de grande taille et sont résolus à l'aide de méthodes itératives et non pas directes. Tout d'abord, nous allons introduire les méthodes de recherche linéaire. Nous aborderons ensuite le problème des moindres carrés non linéaires et la méthode de résolution de Gauss-Newton. Dans le monde de l'optimisation, l'algorithme du gradient conjugué est un outil très puissant pour minimiser une fonction quadratique équivalente au système linéaire à résoudre. Par la suite, nous résoudrons un tel système avec la méthode de Lanczos qui utilise la projection sur un espace particulier. Ensuite, nous allons approximer les valeurs et vecteurs propres d'une matrice à l'aide des paires de Ritz, pour finalement introduire la notion de préconditionnement. Ce chapitre est principalement basé sur les ouvrages de S. Nocedal et S. J. Wright [18] et Y. Saad [19].

2.1 Introduction

Depuis plus d'un demi-siècle, de nombreux d'algorithmes d'optimisation sans contrainte pour les fonctions suffisamment lisses ont été développés. Pour commencer, nous allons illustrer le fonctionnement d'un algorithme d'optimisation dont l'objectif est de minimiser une fonction dite *fonction objectif*

$$f : \mathbb{R}^n \to \mathbb{R}$$

continûment différentiable et suffisamment lisse. Autrement dit, nous cherchons $x^* \in \mathbb{R}^n$ solution de

$$\min_{x \in \mathbb{R}^n} f(x). \tag{2.1}$$

Rappelons quelques notions relatives à ce problème qui seront utilisées dans la suite. Un *minimum global* de (2.1) est un point $x^* \in \mathbb{R}^n$ tel que

$$f(x^*) \leq f(x), \forall\, x \in \mathbb{R}^n.$$

Cependant, la plupart des algorithmes ne sont pas capables de trouver un minimum global car la connaissance de la fonction f, généralement non convexe, est locale. Si f est une fonction convexe, alors un minimum local est un minimum global.

Un *minimum local* de (2.1) est un point $x^* \in \mathbb{R}^n$ pour lequel il existe un voisinage $\mathcal{V}$ de x^* tel que

$$f(x^*) \leq f(x), \forall\, x \in \mathcal{V}.$$

Les deux théorèmes suivants introduisent les conditions d'optimalité que vérifient les solutions de (2.1). Nous parlerons de condition du première ordre et du second ordre lorsque celles-ci impliquent respectivement les dérivées premières et les dérivées secondes de la fonction f.

Théorème 2.1. Condition nécessaire [18]
Si x^* est un minimum local de $f : \mathbb{R}^n \to \mathbb{R}$ continûment différentiable dans le voisinage ouvert $\mathcal{V} \subset \mathbb{R}^n$ de x^*, alors x^* vérifie la condition nécessaire du premier ordre

$$\nabla f(x^*) = 0.$$

De plus, si f est deux fois continûment différentiable sur $\mathcal{V}$, alors x^* vérifie la condition nécessaire du second ordre

$$\nabla^2 f(x^*) \text{ est semi-définie positive.}$$

Théorème 2.2. Condition suffisante [18]
Soit une fonction $f : \mathbb{R}^n \to \mathbb{R}$ deux fois continûment différentiable dans un voisinage ouvert $\mathcal{V} \subset \mathbb{R}^n$ et soit $x^* \in \mathcal{V}$ qui vérifie

$$\nabla f(x^*) = 0 \quad \text{et} \quad \nabla^2 f(x^*) \quad \text{est définie positive,}$$

alors x^* est un minimum local de f.

Les algorithmes que nous utiliserons par la suite sont itératifs et nécessitent un point de départ noté x_0. Le choix de ce point de départ peut provenir de la connaissance de l'application correspondant à la fonction objectif et des données fournies, ce qui permet d'évaluer une bonne estimation de la solution. Autrement, l'algorithme peut estimer un point de départ ou en utiliser un arbitraire.

A partir de ce point de départ, les algorithmes d'optimisation génèrent une suite d'itérés $(x_k)_k$ tels que

$$f(x_{k+1}) < f(x_k), \qquad k = 1, 2, \ldots,$$

jusqu'à ce qu'un certain critère soit satisfait. Cet arrêt peut signifier que la solution a été approchée avec une précision suffisante ou qu'il n'est plus possible de progresser vers une meilleure estimation de la solution que celle obtenue. Il existe deux grandes stratégies de résolution itérative, les méthodes de recherche linéaire et les méthodes de région de confiance. Dans la suite de ce mémoire, seules les premières seront utilisées.

2.2 Méthodes de recherche linéaire

L'algorithme de recherche linéaire calcule une direction de recherche notée p_k. La stratégie de la recherche linéaire consiste à chercher le long de cette direction un nouvel itéré x_{k+1} à partir de l'itéré courant x_k de telle sorte que la valeur de la fonction en le nouvel itéré soit inférieure à celle en l'itéré courant. La direction utilisée doit vérifier $\nabla f(x_k)^T p_k < 0$ ce qui signifie que cette direction est une *direction de descente*. Le nouvel itéré se définit par

$$x_{k+1} = x_k + \alpha_k p_k$$

avec la longueur de pas α_k qui se définit comme la distance parcourue le long de p_k. Elle s'obtient par la résolution du problème d'optimisation

$$\min_{\alpha > 0} f(x_k + \alpha p_k). \tag{2.2}$$

Notons que la résolution de ce problème est approximative car une résolution exacte peut s'avérer coûteuse et est bien souvent non nécessaire. L'algorithme génère quelques longueurs de pas et calcule leurs nouveaux itérés respectifs pour finalement retenir celui approximant grossièrement le minimum de (2.2). Nous parlons alors de recherche linéaire *inexacte* puisqu'elle induit une réduction de f à un coût minime. Pour le nouvel itéré, une nouvelle direction de recherche et une longueur de pas sont calculées et le processus se répète.

2.2.1 Longueur de pas

Pour déterminer une longueur de pas, les algorithmes de recherche linéaire inexacte testent des candidats pour α et s'arrêtent lorsqu'une ou plusieurs conditions sont satisfaites. Dans la suite de ce mémoire, la longueur de pas devra vérifier la *condition d'Armijo* aussi connue comme la première condition de Wolfe sur la décroissance suffisante.

Définition 2.3. Condition d'Armijo [18]
Soit $f : \mathbb{R}^n \to \mathbb{R}$, une fonction différentiable en $x_k \in \mathbb{R}^n$, p_k, une direction de recherche telle que $\nabla f(x_k)^T p_k < 0$ et $\alpha_k \in \mathbb{R}_0^+$, une longueur de pas . La fonction f diminue suffisamment en $x_k + \alpha_k p_k$ par rapport à x_k si

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha \nabla f(x_k)^T p_k \text{ avec } 0 < c_1 < 1.$$

Il existe d'autres conditions comme par exemple celles de *Goldstein*, mais aussi des algorithmes qui utilisent la méthode de *backtracking* pour identifier une longueur de pas adéquate (voir [18]).

2.2.2 Direction de descente

Il existe trois grandes méthodes pour calculer une direction de descente en recherche linéaire. Une première méthode calcule la *direction de plus forte pente* qui est définie à chaque itération par $p_k = -\nabla f(x_k)$. L'avantage est qu'elle ne requiert que la dérivée première de la fonction objectif. Cependant la convergence vers une solution s'avère être lente sur des problèmes complexes.

La direction de descente la plus importante en recherche linéaire est la *direction de Newton* que nous introduisons en nous inspirant du théorème du second ordre de Taylor rappelé ci-dessous.

Théorème 2.4. Théorème de Taylor [18]
Soit $f : \mathbb{R}^n \to \mathbb{R}$, une fonction continûment différentiable et soit $u \in \mathbb{R}^n$. Alors,

$$f(x+u) = f(x) + \nabla f(x+tu)^T u, \tag{2.3}$$

avec $t \in]0,1[$. De plus, si f est deux fois continûment différentiable, alors,

$$f(x+u) = f(x) + \nabla f(x)^T u + \frac{1}{2} u^T \nabla^2 f(x+tu) u \tag{2.4}$$

avec $t \in]0,1[$.

L'approximation de $f(x_k + p)$ par la série de Taylor du second ordre (2.4) est donné par

$$f(x_k + p) \approx f(x_k) + p^T \nabla f(x_k) + \frac{1}{2} p^T \nabla^2 f(x_k) p \overset{not.}{=} m_k(p).$$

En supposant que la matrice hessienne de f est définie positive en x_k, la direction qui minimise $m_k(p)$ et vérifie $\nabla m_k(p) = 0$ est la direction de Newton

$$p_k = -\nabla^2 f(x_k)^{-1} \nabla f(x_k).$$

Pour garantir que p_k soit une direction de descente, la matrice hessienne doit être définie positive. En effet, dans ce cas et si le gradient est non nul,

$$\nabla f(x_k)^T p_k = -\nabla f(x_k)^T \nabla^2 f(x_k)^{-1} \nabla f(x_k) < 0.$$

Si $\nabla^2 f(x_k)$ n'est pas définie positive, il se peut que son inverse n'existe pas. Cependant, dans l'éventualité où l'inverse serait quand même défini, il n'est pas certain que la direction vérifie la propriété de descente, ce qui ne conviendrait pas.

En général, la longueur de pas associée à cette direction est de 1 et peut être ajustée si la réduction de la fonction f n'est pas jugée suffisante. Le taux de convergence des méthodes utilisant une direction de Newton est quadratique. Cependant, la méthode de Newton est une méthode locale et le taux de convergence n'est garanti que lorsque le point de départ est dans le voisinage de la solution.

Le calcul explicite du hessien de f constitue une des faiblesses de ces méthodes vu le coût calculatoire qu'il engendre. Une bonne alternative est l'utilisation des méthodes dites de quasi-Newton. Elles ont la particularité d'avoir un taux de convergence superlinéaire tout en évitant le calcul du hessien. En effet, le véritable hessien est remplacé par la matrice B_k qui, après chaque itération, est mise à jour pour prendre en compte les informations récoltées durant l'itération précédente. Ainsi, la *direction de quasi-Newton* s'écrit

$$p_k = -B_k^{-1} \nabla f(x_k).$$

2.3 Problème des moindres carrés non linéaires

Le problème des moindres carrés non linéaires est un problème d'optimisation sans contrainte de la forme

$$\begin{aligned} \min_{x \in \mathbb{R}^n} f(x) &= \min_{x \in \mathbb{R}^n} \frac{1}{2} \sum_{i=1}^{m} (g_i(x))^2 && (2.5) \\ &= \min_{x \in \mathbb{R}^n} \frac{1}{2}\, g(x)^T g(x) \\ &= \min_{x \in \mathbb{R}^n} \frac{1}{2}\, \|g(x)\|_2^2, && (2.6) \end{aligned}$$

où $m \geq n$ et chaque fonction $g_i : \mathbb{R}^n \to \mathbb{R}$ est continûment différentiable. En assemblant ces fonctions, nous obtenons la *fonction résiduelle* $g : \mathbb{R}^n \to \mathbb{R}^m$ telle que

$$g(x) = [g_1(x), \ldots, g_m(x)]^T$$

qui permet d'écrire (2.6). Ces composantes ne sont pas linéaires, ce qui rend la fonction résiduelle non linéaire. Afin de résoudre ce problème d'optimisation non linéaire, nous allons minimiser itérativement une approximation de Taylor quadratique (2.4) de f au voisinage de l'itéré courant x_k ce qui donne lieu à un système de la forme

$$\nabla^2 f(x_k)\, p = -\nabla f(x_k) \tag{2.7}$$

et peut être résolu à l'aide des méthodes de Newton ou quasi-Newton. En supposant que les m composantes de la fonction résiduelle soient au moins deux fois continûment différentiables, les expressions du gradient et de la

matrice hessienne de (2.7) prennent les formes suivantes

$$\nabla f(x) \;=\; \sum_{i=1}^{m} g_i(x)\,\nabla g_i(x) \;=\; Jac(x)^T\, g(x) \tag{2.8}$$

et

$$\begin{aligned} \nabla^2 f(x) &= \sum_{i=1}^{m} \nabla g_i(x)\,\nabla g_i(x)^T + \sum_{i=1}^{m} g_i(x)\,\nabla^2 g_i(x) \\ &= Jac(x)^T\, Jac(x) + \sum_{i=1}^{m} g_i(x)\,\nabla^2 g_i(x), \end{aligned} \tag{2.9}$$

avec $Jac(x) \in \mathbb{R}^{n\times m}$, la jacobienne du vecteur résiduel telle que

$$Jac(x) = \begin{pmatrix} \nabla g_1(x)^T \\ \vdots \\ \nabla g_m(x)^T \end{pmatrix} = \begin{pmatrix} \dfrac{\partial g_1}{\partial x_1} & \cdots & \dfrac{\partial g_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial g_m}{\partial x_1} & \cdots & \dfrac{\partial g_m}{\partial x_n} \end{pmatrix}.$$

Généralement, le second terme de (2.9) est coûteux à calculer puisqu'il implique des dérivées du second ordre. Pour réduire le coût, la matrice hessienne est approximée par

$$\nabla^2 f(x) \;\approx\; Jac(x)^T\, Jac(x). \tag{2.10}$$

En remplaçant les expressions du gradient (2.8) et de la matrice hessienne (2.10) dans (2.7), le système à résoudre pour chaque approximation quadratique devient

$$Jac(x)^T\, Jac(x)\, p = -Jac(x)^T\, g(x). \tag{2.11}$$

La matrice (2.10) est symétrique mais elle n'est définie positive que si le jacobien $Jac(x)$ est de rang plein. Si c'est le cas et que le gradient est non nul, la direction p sera toujours une direction de descente. Ainsi, il sera toujours possible d'effectuer une recherche linéaire le long de cette direction. La méthode de Gauss-Newton est résumée par l'algorithme 2.1.

Algorithme 2.1 – Les grandes étapes de l'algorithme de Gauss-Newton

Soit x_0 donné

$k \leftarrow 0$

while $(g(x_k) \neq 0)$

1. Calculer $Jac(x_k)$
2. Trouver p_k en résolvant le système :

$$H_k\, p_k = h_k$$

avec $H_k = Jac(x_k)^T\, Jac(x_k)$ et $h_k = -Jac(x_k)^T\, g(x_k)$

3. Calculer une longueur de pas α_k via une recherche linéaire
4. Calculer le nouvel itéré $x_{k+1} = x_k + \alpha_k\, p_k$
5. $k \leftarrow k+1$

end (while)

L'utilisation de cette méthode implique la résolution d'une série de systèmes linéaires symétriques définis positifs de grande taille. Il est donc préférable de les résoudre itérativement. A chaque itération, le système linéaire sera résolu par la méthode du gradient conjugué que nous allons introduire dans la section suivante.

2.4 Méthode du gradient conjugué

La méthode du gradient conjugué (GC) est une des méthodes itératives les plus connues pour résoudre un système linéaire de la forme

$$Ax = b, \tag{2.12}$$

avec la matrice A d'ordre n symétrique définie positive. La résolution de ce système est équivalente à la minimisation de la fonction quadratique strictement convexe

$$\phi(x) = \frac{1}{2} x^T A x - b^T x. \tag{2.13}$$

Nous pouvons définir le résidu du système linéaire par

$$\nabla\phi(x) = Ax - b \overset{not.}{=} r(x).$$

Puisque la fonction ϕ est strictement convexe, son minimum est atteint lorsque son gradient s'annule. Ainsi, les solutions de (2.12) et (2.13) sont identiques et uniques.

2.4.1 La méthode

La méthode du gradient conjugué est un cas particulier de la classe des méthodes de directions conjuguées. Ces méthodes génèrent des directions conjuguées $\{p_0, \ldots, p_{k-1}\}$ par rapport à A telle que

$$p_j^T A p_i = 0, \ \forall\, j \neq i, \tag{2.14}$$

avec $i, j = 0, \ldots, k-1$. De plus, les vecteurs vérifiant la condition (2.14) d'A-conjugaison sont linéairement indépendant. A l'aide de ces directions, nous allons pouvoir générer une suite d'itérés $\{x_0, ..., x_k\}$ via un processus de recherche linéaire. A chaque itération, la longueur de pas sera donnée par

$$\alpha_k = \frac{-r_k^T p_k}{p_k^T A p_k},$$

la solution de (2.2) obtenue par recherche linéaire. Le résidu courant est défini par

$$r(x_k) = Ax_k - b \overset{not.}{=} r_k \tag{2.15}$$

et est orthogonal à toutes les directions précédentes

$$r_k^T p_i = 0,$$

avec $i = 0, \ldots, k-1$. Pour obtenir la nouvelle direction conjuguée p_k, la méthode du GC n'utilise que la direction précédente p_{k-1} ce qui permet d'éviter de stocker les directions. De plus, p_k est automatiquement conjuguée aux directions précédentes en initialisant la première direction p_0 avec le résidu initial r_0. Il est ainsi possible de considérer la méthode du gradient conjugué comme une méthode itérative. Les autres directions conjuguées sont chacune une combinaison linéaire de la direction précédente et de la direction de plus forte pente

$$p_k = -r_k + \beta_k p_{k-1} \tag{2.16}$$

avec β_k choisi de sorte que la direction précédente et celle générée soient A-conjuguées. En effet, la relation (2.14) avec p_{k-1} et (2.16) permet d'obtenir

$$\beta_k = \frac{r_k^T A p_{k-1}}{p_{k-1}^T A p_{k-1}}. \tag{2.17}$$

2.4.2 L'algorithme

L'algorithme 2.2 décrit la méthode ci-dessus pour un point de départ x_0 donné. Cependant les expressions de (2.15) et (2.17) ne correspondent pas à celles données précédemment. En effet, pour réduire le coût calculatoire de l'algorithme, il est possible de réécrire ces expressions différemment et ainsi éviter la redondance du produit matrice-vecteur $A\,p_k$. Les étapes du raisonnement permettant d'obtenir l'algorithme 2.2 sont disponibles dans [20].

Algorithme 2.2 – Gradient conjugué [18]

Soit x_0 donné

$r_0 \leftarrow A\,x_0 - b$, $p_0 \leftarrow -r_0$, $k \leftarrow 0$

while ($r_k \neq 0$)

$$\begin{aligned}
\alpha_k &\leftarrow \frac{r_k^T\, r_k}{p_k^T\, A\, p_k} \\
x_{k+1} &\leftarrow x_k + \alpha_k\, p_k \\
r_{k+1} &\leftarrow r_k + \alpha_k\, A\, p_k \\
\beta_{k+1} &\leftarrow \frac{r_{k+1}^T\, r_{k+1}}{r_k^T\, r_k} \\
p_{k+1} &\leftarrow -r_{k+1} + \beta_{k+1}\, p_k \\
k &\leftarrow k+1
\end{aligned}$$

end (while)

A la fin de l'algorithme, seuls les vecteurs x, r et p des deux dernières itérations sont en mémoire. La méthode du GC n'altère pas les coefficients de la matrice contrairement aux méthodes de résolution qui utilisent de la factorisation. Cependant, dans la suite, nous allons détailler certains de ses inconvénients.

2.4.3 Propriétés de la méthode

La méthode du gradient conjugué appartient à la famille des méthodes de résolution des sous-espaces de Krylov. Elles ont pour particularité de minimiser la fonction objectif sur ces sous-espaces.

Définition 2.5. Sous-espace de Krylov [21]
Soit une matrice $A \in \mathbb{R}^{n\times n}$ non singulière et symétrique et $b \in \mathbb{R}^n$ un vecteur normé. Le sous-espace de Krylov de dimension $k \leq n$ lié à la matrice A est défini par

$$\mathcal{K}(A,b,k) = Span\{b, Ab, \ldots, A^{k-1}b\}.$$

Le gradient conjugué est la réalisation d'une technique de projection orthogonale sur le sous-espace de Krylov $\mathcal{K}(A,r_0,k)$. Reprenons en quelques lignes les propriétés importantes du gradient conjugué, provenant de [18],

$$\begin{aligned}
p_k^T\, A\, p_i &= 0, \\
r_k^T\, p_i &= 0, \\
r_k^T\, r_i &= 0,
\end{aligned}$$

$\forall\, i = 0, \ldots, k-1$,

$$r_k^T A\, p_i \quad = 0,$$

$\forall\, i = 0, \ldots, k-2$,

$$r_{i+1} = r_i + \alpha_i A\, p_i,$$

$\forall\, i = 0, \ldots, k.$

Pour ce qui est de la convergence, les deux théorèmes 2.6 et 2.7 résument les principales propriétés en arithmétique exacte.

Théorème 2.6. [18, p. 115]
Si la matrice $A \in \mathbb{R}^{n\times n}$ possède $r \leq n$ valeurs propres distinctes, alors l'algorithme du gradient conjugué convergera en la solution exacte $x_k = x^*$ avec $k \leq r$.

Théorème 2.7. [22, p. 18]
Soit A, une matrice d'ordre n symétrique définie positive, $\mathcal{K}(A)$, le nombre de conditionnement en norme euclidienne de la matrice A est défini par

$$\mathcal{K}(A) = || A^{-1} ||_2 \, || A ||_2 = \frac{\lambda_{max}}{\lambda_{min}}.$$

L'erreur e_k des itérés générés par l'algorithme 2.2 vérifie

$$\frac{|| x_k - x_* ||_A}{|| x_0 - x_* ||_A} = \frac{|| e_k ||_A}{|| e_0 ||_A} \leq 2 \left(\frac{\sqrt{\mathcal{K}(A)} - 1}{\sqrt{\mathcal{K}(A)} + 1} \right)^k.$$

Nous observons que le nombre d'itérations n'excède pas la dimension de la matrice A en arithmétique exacte. De plus, la totalité du spectre influence la convergence. Au plus les valeurs propres sont rassemblées, autrement dit *clusterisées*, au plus la convergence est rapide. De ce dernier résultat, nous déduisons qu'en réduisant le nombre de conditionnement de A, la convergence de la méthode est améliorée. Ce raisonnement va être au cœur de l'idée du préconditionnement introduit par la suite.

Remarque [23, p. 146] : En arithmétique exacte, la méthode du gradient conjugué converge en un nombre fini d'étapes. En pratique, le phénomène de *breakdown* correspond à l'accumulation d'erreurs d'arrondi qui peuvent provoquer la destruction de l'A-orthogonalité des directions de descente. En conséquence, lors du calcul des coefficients α et β, des divisions par zéro peuvent apparaitre. Il faut donc recourir à un processus de stabilisation qui conserve les propriétés de la méthode.

2.5 Théorie de Lanczos

Il existe une autre approche que celle du gradient conjugué pour résoudre le système (2.12). La méthode de Lanczos est construite sur la projection sur un sous-espace particulier à l'aide de vecteurs particuliers. Elle permet d'approximer les valeurs propres de A et plus particulièrement les valeurs propres extrêmes sans beaucoup d'effort à l'aide de l'approximation de Ritz.

2.5.1 Problème aux valeurs propres et paires de Ritz

De nouveau, supposons que nous avons une matrice A d'ordre n symétrique définie positive. Nous souhaitons estimer les paires propres (λ, v) de A sur un sous-espace donné $\mathcal{S} \in \mathbb{R}^n$. La symétrie de A garantit que les valeurs propres sont réelles et qu'il existe une base orthonormale de vecteurs propres. L'information de Ritz va être utilisée pour approximer les paires propres de A.

Définition 2.8. Résidu d'une paire propre [21]
Soit $z \in \mathcal{S} \subset \mathbb{R}^n$ et $\theta \in \mathbb{R}_0^+$, le résidu de l'approximation par (θ, z) de la paire propre (λ, v) est défini par le vecteur

$$\hat{r} = Az - \theta z. \tag{2.18}$$

Supposons que $Q = \{q_1, \ldots, q_k\}$ est une base d'un sous-espace de $\mathbb{R}^n$ avec $k \leq n$. Nous allons utiliser l'*approche Galerkin* de sorte que

$$z \in \text{Span}\{q_1, \ldots, q_k\} \text{ et } \theta \in \mathbb{R}, \tag{2.19}$$

pour trouver une paire (θ, z) qui approxime une paire propre de A. Cette approche impose que le résidu de la paire utilisée soit orthogonal aux vecteurs de la base Q ,

$$Q^T \hat{r} = Q^T (Az - \theta z) = 0. \tag{2.20}$$

A partir de (2.19), nous pouvons écrire

$$z = Qy \tag{2.21}$$

avec $y \in \mathbb{R}^k$. En remplacant (2.21) dans (2.20), nous obtenons

$$Q^T (AQy - \theta Qy) = 0, \tag{2.22}$$

qui peut être simplifiée si Q est choisie comme étant une base orthonormale. En effet, (2.22) devient

$$Q^T AQy - \theta y = 0 \tag{2.23}$$

avec $Q^T AQ \in \mathbb{R}^{k \times k}$ symétrique. En utilisant Q, nous avons pu ramener un problème aux valeurs propres de taille n à un problème de taille k avec y comme inconnue. A présent, nous pouvons donner la définition d'une paire de Ritz.

Définition 2.9. Paire de Ritz [21]
Les solutions du problème aux valeurs propres

$$Q^T AQy = \theta y \tag{2.24}$$

sont les paires primitives de Ritz (θ_i, y_i), $i = 1, \ldots, k$, de la matrice A associé au sous-espace Span $\{q_1, \ldots, q_k\}$. Les paires de Ritz (θ_i, z_i) sont définies à partir des paires primitives telles que

$$z_i = Qy_i \tag{2.25}$$

pour $i = 1, \ldots, k$.

De plus, les vecteurs z_i sont orthonormaux et A-conjugués [21, p. 15].

2.5.2 Algorithme de Lanczos

Tout comme la méthode du gradient conjugué, la méthode de Lanczos fait partie des méthodes de Krylov. L'approximation de la solution d'un système d'équations linéaires $Ax = b$ à l'étape k du gradient conjugué est

$$x_k \in x_0 + \mathcal{K}(A, r_0, k),$$

où x_0 est un vecteur initial dont le résidu associé est $r_0 = Ax_0 - b$ et $\mathcal{K}(A, r_0, k)$ est un sous-espace de Krylov. La solution x_k est construite à l'aide du vecteur initial et d'une base du sous-espace de Krylov au sein de laquelle une projection orthogonale est réalisée. L'algorithme de Lanczos permet la construction de cette base orthonormale à partir d'un vecteur q_1 donné.

Algorithme 2.3 – Méthode de Lanczos pour les problèmes aux valeurs propres symétriques [21]

Soit q_1 donné

$r_0 \leftarrow q_1$, $\eta_0 \leftarrow 1$, $q_0 \leftarrow 0$, $k \leftarrow 0$

while ($\eta_k \neq 0$)

$$\begin{aligned} q_{k+1} &\leftarrow \frac{r_k}{\eta_k} \\ k &\leftarrow k+1 \\ \sigma_k &\leftarrow q_k^T A\, q_k \\ r_k &\leftarrow (A - \sigma_k I_n)\, q_k - \eta_{k-1}\, q_{k-1} \\ \eta_k &\leftarrow \|r_k\|_2 \end{aligned}$$

end (while)

Une fois l'algorithme 2.3 terminé, nous obtenons les vecteurs $q_1, \ldots, q_{k+1}$ appelés *vecteurs de Lanczos*. Ceux-ci forment la base souhaitée puisque les k premiers vecteurs permettent de construire la matrice $Q_k = [q_1, \ldots, q_k]$.

Proposition 2.10. [24]
Supposons que les vecteurs de Lanczos $q_1, \ldots, q_k$ forment les colonnes de la matrice $Q_k \in \mathbb{R}^{n\times k}$. Les éléments $\sigma_1, \ldots, \sigma_k$ et $\eta_1, \ldots, \eta_{k-1}$ fournis par l'algorithme de Lanczos 2.3 permettent de construire la matrice tridiagonale

$$T_k = \begin{pmatrix} \sigma_1 & \eta_1 & \ldots & 0 \\ \eta_1 & \sigma_2 & \ldots & 0 \\ \vdots & \ldots & \ddots & \eta_{k-1} \\ 0 & 0 & \eta_{k-1} & \sigma_k \end{pmatrix}. \tag{2.26}$$

Ces deux matrices vérifient les égalités suivantes

$$A\, Q_k = Q_k\, T_k + \eta_k\, q_{k+1}\, e_k^T \tag{2.27}$$

$$T_k = Q_k^T\, A\, Q_k. \tag{2.28}$$

Cette dernière relation est appelée relation fondamentale de Lanczos.

L'algorithme de Lanczos calcule à chaque itération une tridiagonalisation partielle de la matrice A ce qui permet d'énoncer un lien étroit entre les valeurs propres de A et de T_k. L'information sur les valeurs propres extrêmes

du spectre de A apparaît assez tôt au cours des itérations, ce qui rend la méthode très utile lorsqu'il s'agit d'avoir quelques plus grandes ou plus petites valeurs propres avec les vecteurs propres associés. Dans un premier temps, les paires propres de la matrice T_k sont calculées, elles correspondent aux paires primitives de Ritz (θ_i, y_i), $i = 1, \ldots, k$. Ensuite les paires de Ritz sont calculées par $z_i = Q_k y_i$ pour obtenir les paires (θ_i, z_i), $i = 1, \ldots, k$.

Approximation de Ritz

Nous allons calculer l'erreur d'approximation des paires propres de A directement liée à la convergence des paires de Ritz. En multipliant l'équation (2.28) par y_i, nous obtenons

$$A\,Q_k\,y_i = Q_k\,T_k\,y_i + \eta_k\,q_{k+1}\,e_k^T\,y_i.$$

Nous pouvons déduire de (2.28) et (2.24) que $T_k\,y_i = \theta_i\,y_i$ et en utilisant $z_i = Q\,y_i$, nous obtenons

$$A\,z_i - \theta_i\,z_i = (e_k^T\,y_i)\,\eta_k\,q_{k+1}$$

dont le terme de droite correspond au résidu (2.18) Enfin, la norme euclidienne de ce résidu peut s'écrire

$$\|\,\hat{r}\,\|_2 \;=\; \|\,A\,z_i - \theta_i\,z_i\,\|_2 \;=\; \eta_k\,|\,e_k^T\,y_i\,| \tag{2.29}$$

et va nous permettre d'évaluer l'erreur lors de l'approximation des paires propres.

Lanczos pour les systèmes linéaires

L'algorithme de Lanczos permet d'approximer la solution du système linéaire $Ax = b$ comme le montre le théorème suivant.

Théorème 2.11. [25, p. 490]
Soit la matrice $Q_k \in \mathbb{R}^{n\times k}$ dont les colonnes sont les vecteurs de Lanczos obtenus après k itérations de l'algorithme 2.3. Supposons x_0 un vecteur initial arbitraire, alors

$$x_k = x_0 + Q_k\,w_k, \tag{2.30}$$

où w_k est la solution de

$$(Q_k^T\,A\,Q_k)\,w_k = Q_k^T\,(b - A\,x_0),$$

minimise la fonction $\dfrac{1}{2}x^TAx - b^Tx$ sur $x_0 +$ Span $\{q_1, \ldots, q_k\}$. De plus, si $k = n$,la minimisation se fait sur $\mathbb{R}^n$ et x_k est la solution de $Ax = b$.

En pratique, ce résultat permet de calculer w_k en résolvant

$$T_k\,w_k = Q_k^T\,(b - A\,x_0),$$

et en utilisant (2.28). La solution de $A\,x = b$ s'approxime ensuite par (2.30).

Remarque [26] : Théoriquement, l'algorithme de Lanczos est construit de telle sorte que les vecteurs de Lanczos sont orthonormalisés. Malheureusement, en précision finie, ils tendent à perdre leur propriété d'orthogonalité, et ce, à cause de l'accumulation d'erreurs d'arrondis. Par conséquent, les valeurs de Ritz sont incorrectes puisque la matrice tridiagonale est impactée par ces erreurs. En pratique, il est donc nécessaire de recourir à une réorthogonalisation complète afin d'éviter la perte d'orthogonalité. Nous utiliserons la méthode de Gram-Schmidt modifiée (voir [24, p. 255]) pour pallier à ce problème.

2.5.3 Relation entre l'algortihme de Lanczos et l'algorithme du gradient conjugué

L'objectif est de mettre en évidence la relation qu'entretiennent l'algorithme de Lanczos et celui du gradient conjugué. Il peut être utile d'obtenir la matrice T_k construite à partir des k itérations de l'algorithme de Lanczos depuis les coefficients générés par l'algorithme du gradient conjugué. Le théorème 2.12 exprime les coefficients de Lanczos en fonction de ceux du gradient conjugué. La correspondance entre les paires de coefficients (α, β) et (σ, η) provenant des deux algorithmes est détaillée par Y. Saad [19, p. 202].

Proposition 2.12. [26, p. 51]
Soient η_j, σ_j les coefficients définis dans l'algorithme de Lanczos et α_j, β_j les coefficients de l'algorithme du gradient conjugué. Le lien entre les paires de coefficients est donné par

$$\sigma_{j+1} = \begin{cases} \dfrac{1}{\alpha_0} & \text{pour } j = 0, \\ \dfrac{1}{\alpha_j} + \dfrac{\beta_{j-1}}{\alpha_{j-1}} & \text{pour } j > 0 \end{cases} \qquad \text{et} \qquad \eta_{j+1} = \frac{\sqrt{\beta_{j-1}}}{\alpha_{j-1}}.$$

En pratique, cette proposition sera utile lorsque nous devrons résoudre un système linéaire symétrique défini positif par l'algorithme du gradient conjugué et que nous devrons ensuite disposer de la matrice tridiagonale T pour obtenir l'information de Ritz.

2.6 Méthode du gradient conjugué préconditionné

La convergence de la méthode du gradient conjugué dépend de la distribution spectrale de la matrice du système. Elle peut être ralentie par un mauvais nombre de conditionnement de la matrice A ou une mauvaise distribution des valeurs propres de A. Les système linéaires $Ax = b$ sont dits mal conditionnés si une petite variation de A et de b entraîne une grande variation de la solution x. Le processus de préconditionnement consiste à effectuer un changement de variables pour passer de x à $\hat{x}$ en utilisant une matrice non singulière C telle que

$$\hat{x} = Cx \tag{2.31}$$

Le système linéaire défini par (2.12) devient

$$(C^{-T} A C^{-1})\hat{x} = C^{-T} b \tag{2.32}$$

et la fonction à minimiser définie en (2.13) devient

$$\hat{\phi}(\hat{x}) = \frac{1}{2}\hat{x}^T (C^{-T} A C^{-1})\hat{x} - (C^{-T} b)^T \hat{x}. \tag{2.33}$$

Le taux de convergence ne dépend plus de A mais de $C^{-T} A C^{-1}$. Ainsi, la distribution des valeurs propres de la nouvelle matrice va permettre d'améliorer le taux de convergence. La matrice C doit être choisie de sorte que le nombre de conditionnement de la matrice $C^{-T} A C^{-1}$ soit meilleur que celui de A. La matrice de changement de variable peut aussi être choisie de sorte que les valeurs propres de $C^{-T} A C^{-1}$ soient mieux clusterisées que celles de A.

L'algorithme du gradient conjugué peut être appliqué pour résoudre (2.32). En adaptant l'initialisation et la boucle de ce dernier, les expressions utilisant la variable $\hat{x}$ peuvent être réexprimées de sorte que la solution finale soit

celle du système (2.12). Ainsi, nous obtenons l'algorithme 2.4 qui est l'algorithme du gradient conjugué non préconditionné (GCP). Celui-ci n'utilise pas explicitement la matrice C mais la matrice $M = C^T C$ qui est symétrique définie positive par construction.

Algorithme 2.4 – Gradient conjugué préconditionné [18]

Soit x_0 et M donnés

$r_0 \leftarrow A x_0 - b$, résoudre $M z_0 = r_0$, $p_0 \leftarrow -z_0$, $k \leftarrow 0$

while ($r_k \neq 0$)

$$\begin{aligned}
\alpha_k &\leftarrow \frac{r_k^T z_k}{p_k^T A p_k} \\
x_{k+1} &\leftarrow x_k + \alpha_k p_k \\
r_{k+1} &\leftarrow r_k + \alpha_k A p_k \\
&\text{résoudre } M z_{k+1} = r_{k+1} \\
\beta_{k+1} &\leftarrow \frac{r_{k+1}^T z_{k+1}}{r_k^T z_k} \\
p_{k+1} &\leftarrow -z_{k+1} + \beta_{k+1} p_k \\
k &\leftarrow k+1
\end{aligned}$$

end (while)

Le GCP est plus coûteux que GC puisqu'un système linéaire $M z_{k+1} = r_{k+1}$ est à résoudre en plus à chaque itération. Dans le chapitre suivant, les préconditionneurs que nous allons construire approximent l'inverse la matrice de A.

Notons finalement qu'il n'existe pas de technique universelle pour développer un préconditionneur, mais il est préférable de respecter certaines règles pour que celui-ci soit efficace [27] :

1. M doit être symétrique défini positif ou de manière équivalente que C doit être non singulière ;
2. $\mathcal{K}(MA) < \mathcal{K}(A)$;
3. les valeurs propres de MA doivent être plus clusterisées que celles de A ;
4. le préconditionneur ne doit pas être coûteux en termes de calculs et d'application.

2.7 Conclusion

Ce chapitre a permis de mettre en place l'ensemble des outils nécessaires à la résolution de systèmes linéaires symétriques définis positifs par des méthodes de recherche linéaire. Nous avons abordé l'algorithme du gradient conjugué et sa version préconditionnée. De plus, la théorie de Lanczos et l'information de Ritz vont nous permettre d'approximer les paires propres de la matrice du système à résoudre dans les deux chapitres qui vont suivre. Par la suite, nous allons combiner résolution efficace et obtention de l'information de Ritz. Cela sera possible grâce au lien que nous avons décrit entre les coefficients des algorithmes du gradient conjugué et de Lanczos. Le chapitre suivant a pour objectif de mettre en place trois instances d'une classe de préconditionneurs spécifique. L'information pour les construire provient des différentes étapes de ces deux algorithmes.

Chapitre 3

Précondtionneurs à mémoire limitée

Le taux de convergence des méthodes de type gradient conjugué dépend de la distribution des valeurs propres de la matrice du système. L'utilisation d'une matrice de préconditionnement peut permettre, si elle est bien construite, d'améliorer cette distribution. Ce chapitre proprose une classe de préconditionneurs dénomée préconditionneurs à mémoire limitée, en anglais, *Limited Memory Preconditioner* (LMP). Nous commencerons par en donner la formulation générale ainsi que les propriétés principales afin de montrer que ce type de préconditionneurs est peu coûteux à construire et à stocker. Trois instances seront mises en avant, il s'agit du spectral-LMP, du Ritz-LMP et du quasi-Newton-LMP. Les deux premières utilisent l'information spectrale tandis que la dernière est construite à partir de directions de descente générées par l'algorithme du gradient conjugué. Le Ritz-LMP fut proposé pour la première fois par J. Tshimanga et ses co-auteurs. Ensuite, à l'aide d'expérimentations numériques, nous observerons l'efficacité de ces préconditionneurs. Nous donnerons une brève comparaison ainsi que des illustrations permettant de comparer leur performance. Le lecteur pourra se reporter à la thèse de J. Tshimanga [21] pour les preuves de divers résultats ainsi qu'aux articles [27] et [28] qui y sont relatifs.

3.1 Formulation générale du LMP

Dans cette section, nous allons introduire la formulation générale des préconditionneurs à mémoire limitée ainsi que les propriétés principales qu'ils vérifient. Dans la suite, pour désigner ces préconditionneurs, nous utiliserons la notation LMP pour Limited Memory Preconditioner. Nous cherchons à résoudre un système linéaire $Ax = b$ avec une matrice A d'ordre n symétrique définie positive. Le chapitre précédent a permis de rappeler comment résoudre ce système avec des méthodes de type gradient conjugué. Nous avons vu qu'il est souvent utile de recourir à du préconditionnement pour accélérer ces algorithmes. Comme dit précédemment, un bon préconditionneur est un préconditionneur qui approxime l'inverse de A.

Supposons que nous disposons d'un préconditonneur M appelé *préconditionneur de premier niveau*, qui clusterise un bon nombre de valeurs propres autour de 1. L'idée générale des LMP est d'améliorer l'efficacité de ce premier niveau de préconditionnement. En effet, les LMP sont considérés comme un *second niveau de préconditionnement* puisqu'ils ont pour but de collecter de l'information que le premier niveau n'a pas pris en compte et qui ralentit la convergence. Ainsi nous obtenons la définition suivante.

Définition 3.1. Préconditionneur à mémoire limitée [28]
Soit A et M deux matrices d'ordre n symétriques définies positives et I_n la matrice identité d'ordre n.

Supposons que S soit une matrice $n \times l$ de rang l, avec $l \leq n$. La matrice symétrique

$$K_l = [I_n - S(S^T A S)^{-1} S^T A] M [I_n - A S (S^T A S)^{-1} S^T] + S (S^T A S)^{-1} S^T \tag{3.1}$$

est appelée préconditionneur à mémoire limitée.

Notons que l'inverse de K_l peut être calculé via la formule de Sherman-Morrison (voir [29]). La formulation de K_l ne requiert pas que le préconditionneur soit stocké comme une matrice mais bien que seules les colonnes de S le soient, d'où la notion de mémoire limitée. L'indice associé au préconditionneur fait référence au nombre de vecteurs de la matrice S qui ont permis sa construction. Nous avons omis d'indicer la matrice S par son nombre de colonnes afin de ne pas alourdir l'écriture.

Nous allons évoquer les propriétés principales des LMP sous cette formulation avant de nous intéresser aux trois formulations particulières. Nous faisons l'hypothèse que les colonnes de $S = [s_1, \ldots, s_l]$ sont linéairement indépendantes. Plus loin, les propriétés d'orthogonalité, d'invariance par rapport à A et de conjugaison par rapport à A influenceront la formulation (3.1).

3.1.1 Propriétés générales

L'ensemble des propriétés importantes des LMP vont être parcourues, et au fur et à mesure, les différentes exigences d'un bon préconditionneur vont être vérifiées. La première exigence concerne l'invariance de la matrice K_l sous le changement de base des vecteurs de la matrice S, comme le montre la proposition 3.2.

Proposition 3.2. [28, p.3]
Soit la matrice $Z \in \mathbb{R}^{n \times l}$ définie par $Z = SX$ avec X une matrice carrée inversible d'ordre l.
Alors l'égalité suivante est vérifiée

$$S (S^T A S)^{-1} S^T = Z (Z^T A Z)^{-1} Z^T$$

et a pour conséquence que

$$K_l = [I_n - Z(Z^T A Z)^{-1} Z^T A] M [I_n - A Z (Z^T A Z)^{-1} Z^T] + Z (Z^T A Z)^{-1} Z^T .$$

En pratique, en analyse numérique, l'inverse de la matrice A n'est pas calculé mais il est approximé. Toutefois, il est intéressant de remarquer que lorsque le nombre de colonnes de S équivaut à la dimension de A, alors K_n correspond à A^{-1}. Le préconditionneur LMP satisfait les propriétés structurelles : il est symétrique par construction et défini positif (voir annexe A de [27]). Le théorème 3.3 détaille l'effet du préconditionnement de (3.1) sur A et spécifie que les l valeurs propres de $K_l A$ valent 1 et que le reste du spectre ne s'étend pas par rapport à celui de A.

Théorème 3.3. [21, p. 38]
Soit A, une matrice d'ordre n symétrique définie positive et K_l défini par (3.1).
Si les nombres réels positifs $\lambda_1, \ldots, \lambda_n$ et $\mu_1, \ldots, \mu_n$ sont organisés de manière croissante et correspondent respectivement aux valeurs propres de A et $K_l A$, alors $K_l A$ est une matrice non défective et il existe un indice m, $1 \leq m \leq n - l$, *tel que*

$$\begin{cases} \mu_j = 1 & \text{pour } j \in \{m, \ldots, m+l\}, \\ \lambda_1 \leq \mu_j \leq \lambda_n & \text{pour } j \in \{1, \ldots, m-1\} \cup \{m+l+1, \ldots, n\}. \end{cases}$$

Les figures 3.1 sont inspirées de [21] et donnent les représentations schématiques de cette propriété pour trois distributions spectrales de A différentes : inférieure à 1, supérieure à 1 ou de part et d'autre de 1.

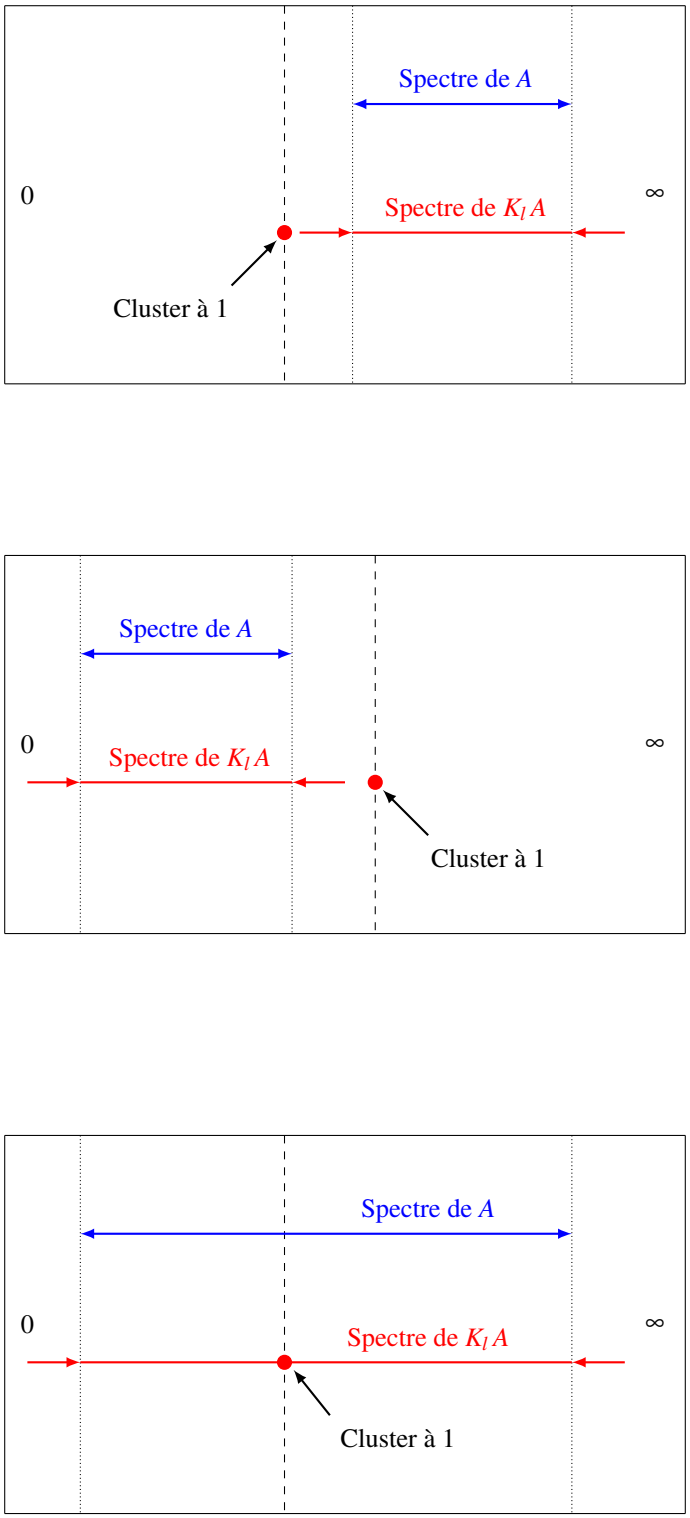


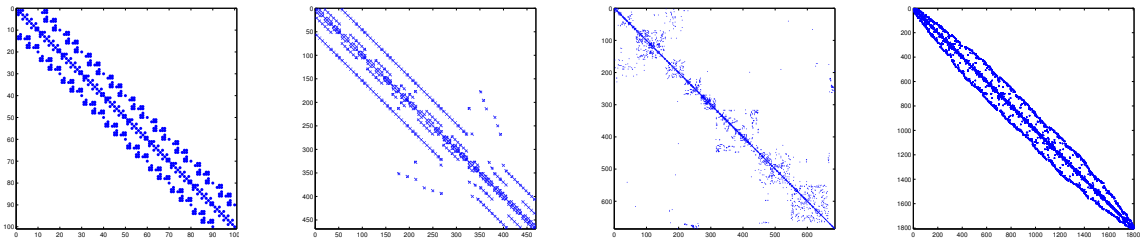
FIGURE 3.1 – Représentations schématiques de l'effet du préconditionneur K_l sur le spectre de A.

Dans certains cas, il se peut que la matrice A possède déjà des valeurs propres proches de 1, par exemple grâce à un préconditionneur de premier niveau. Lors de l'application du second niveau de préconditionnement par LMP, l'objectif est de grouper un peu plus le spectre autour de 1 et ce en conservant le cluster déjà existant. Dans sa thèse, J. Tshimanga a démontré que la multiplicité de la valeur propre 1 ne diminue pas lorsque le LMP est appliqué à A [21, p. 42].

3.1.2 Illustrations numériques

Afin d'illustrer les propriétés des LMP, nous allons considérer quatre matrices symétriques définies positives de la collection *Harwell-Boeing Sparse Matrix Collection* [30]. Ces matrices vont permettre d'illustrer l'action spectrale du préconditionneur LMP. Notons que l'objectif ici n'est pas d'explorer les effets sur toutes les distributions propres possibles mais uniquement de l'illustrer sur des choix particuliers. En effet, les trois premières matrices ont respectivement un spectre inférieur à 1, supérieur à 1 ou de part et d'autre de 1. Une quatrième possède un spectre avec une clusterisation à 1 déjà existante. Le tableau suivant reprend les caractéristiques importantes des matrices tests et la figure 3.2 permet de les visualiser. Le calcul du conditionnement a été obtenu via la fonction `cond` de Matlab.

Nom	Matrice 1 `nos4`	Matrice 2 `nos5`	Matrice 3 `BUS685`	Matrice 4 `BCSSTK14`
dimension n	100	468	685	1806
cond(A)	$1.57\,10^{3}$	$1.10\,10^{4}$	$4.23\,10^{5}$	$1.19\,10^{10}$
λ_{max}	$5.38\,10^{-4}$	$5.29\,10^{1}$	$6.19\,10^{-2}$	1
λ_{min}	$8.49\,10^{-1}$	$5.82\,10^{5}$	$2.62\,10^{4}$	$1.19\,10^{10}$



(a) Matrice 1 (b) Matrice 2 (c) Matrice 3 (d) Matrice 4

FIGURE 3.2 – Représentation des éléments non nuls des matrices.

Procédure : Pour toutes les matrices nous avons généré $l = n/4$ vecteurs aléatoires $s_1, \ldots, s_l$ linéairement indépendants. Ensuite, la matrice de préconditionnement K_l est construite dans le but de comparer la distribution des valeurs propres de A et de $K_l A$.

Les résultats obtenus pour les quatre matrices choisies sont illustrés sur les figures 3.3 à 3.6. Des lignes horizontales ont été tracées afin de repérer les valeurs extrêmes du spectre. La double flèche représente l'étendue du spectre et la ligne rouge correspond à la valeur propre 1. Nous observons la clusterisation des valeurs propres autour de 1 après conditionnement sur les trois premières figures et ce peu importe la distribution du spectre de A. De plus, il n'y a pas d'expansion du spectre. La figure 3.6 indique qu'avant le préconditionnement, une partie du spectre était clusterisée autour de 1. L'application du préconditionneur n'a pas détruit cette clusterisation déjà existante et a permis de l'agrandir. Nous pouvons conclure que les diverses propriétés décrivant l'effet du LMP sur le spectre sont vérifiées.

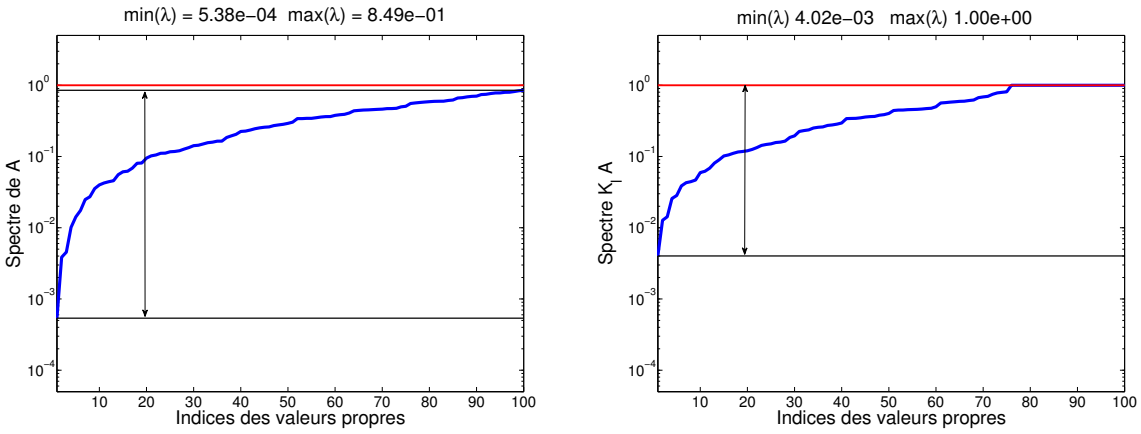


FIGURE 3.3 – Distribution spectrale relative à la matrice 1.

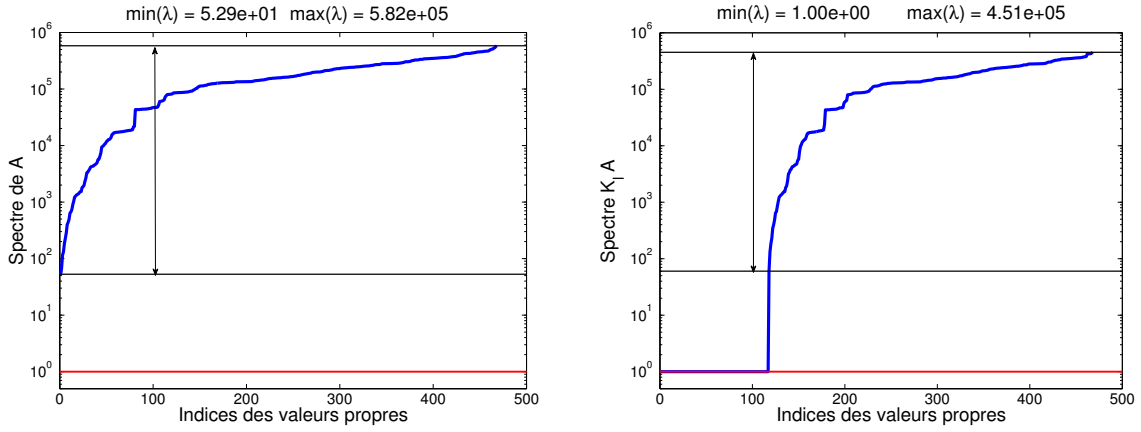


FIGURE 3.4 – Distribution spectrale relative à la matrice 2.

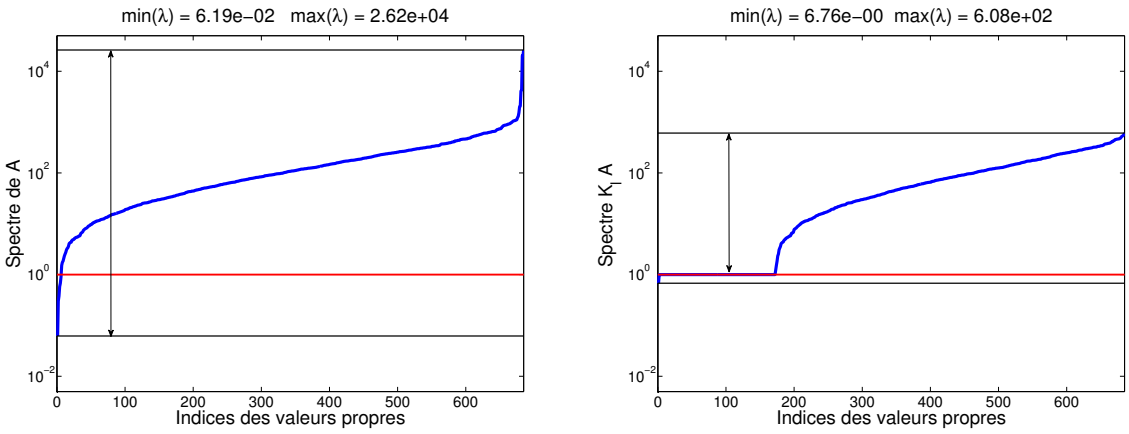


FIGURE 3.5 – Distribution spectrale relative à la matrice 3.

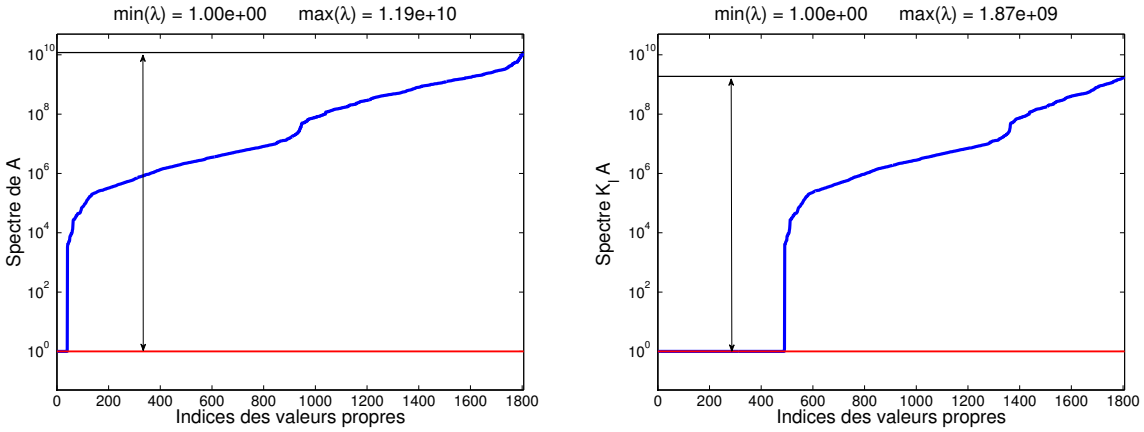


FIGURE 3.6 – Distribution spectrale relative à la matrice 4.

3.1.3 Forme factorisée et empilement

Nous allons introduire l'expression de la forme factorisée du préconditionneur LMP (3.1). Bien que la méthode du gradient conjugué préconditionné ne nécessite pas une forme particulière du préconditionneur, cette nouvelle formulation est particulièrement pratique pour combiner les LMP dans certains contextes comme celui de l'utilisation de la méthode de Gauss-Newton.

Théorème 3.4. [28, p. 13]
Soit $M = LL^T$ avec $L^T \in \mathbb{R}^{n\times n}$ et $S^T AS = R^T R$ avec $R \in \mathbb{R}^{l\times l}$, une factorisation de M et $S^T AS$ respectivement. Alors la forme factorisée de la matrice LMP (3.1) *s'écrit $K_l = G_l G_l^T$, avec G_l donné par*

$$G_l = L - SR^{-1}R^{-T}S^T AL + SR^{-1}X^{-T}S^T L^{-T}, \tag{3.2}$$

où $S^T L^{-T} L^{-1} S = X^T X$, avec $X \in \mathbb{R}^{l\times l}$, une factorisation de $S^T L^{-T} L^{-1} S$.

Néanmoins, cette factorisation n'est pas unique puisque toutes les formes factorisées sont équivalentes à (3.2). En effet, prenons une autre factorisation de K_l, $\tilde{G}_l = G_l Q$, avec Q une matrice orthogonale d'ordre n. Nous avons que $\tilde{G}_l \tilde{G}_l^T = G_l G_l^T$ est équivalent à $G_l^{-1}\tilde{G}_l (G_l^{-1}\tilde{G}_l)^T = I_n$. En pratique, la factorisation de Cholesky de K_l est coûteuse en terme de calcul et de stockage et n'est pas toujours recommandée. Les LMP ont pour avantage d'avoir une forme factorisée qu'il est possible de construire sans passer par la forme générale K_l. Cependant, (3.2) repose sur la forme factorisée du préconditionneur de premier niveau. Si celle-ci n'est pas disponible, il est possible de l'obtenir via certains procédés comme la factorisation de Cholesky. D'un point de vue coût, le calcul de la factorisation de M sera aussi conséquent que la factorisation de $K_l = G_l G_l^T$ à moins que l soit relativement petit [28].

Empilement

Dans l'algorithme du gradient conjugué préconditionné, il est possible de travailler avec plusieurs préconditionneurs en les empilant. Ce processus est connu sous le nom de *stacking*. Supposons que nous disposons de la matrice A d'ordre n symétrique définie positive d'un système à préconditionner ainsi que de p matrices de préconditionnement, $p \geq 2$, symétriques définies positives. Ces matrices

$$\begin{aligned} K_{l_1}^{(1)} &= G_{l_1}^{(1)} G_{l_1}^{(1)^T}, \\ &\vdots \\ K_{l_p}^{(p)} &= G_{l_p}^{(p)} G_{l_p}^{(p)^T} \end{aligned}$$

sont disponibles simultanément et peuvent être utilisées pour obtenir un préconditionneur K_{comp} de façon multiplicative

$$K_{comp} = G_{l_p}^{(p)^T} \ \ldots \ G_{l_2}^{(2)^T} K_{l_1}^{(1)} G_{l_2}^{(2)} \ \ldots \ G_{l_p}^{(p)}. \tag{3.3}$$

Le résultat de cette composition multiplicative de matrices définies positives est une matrice définie positive. De plus, pour l'utiliser comme préconditionneur du gradient conjugué préconditionné, la symétrie doit être garantie ce qui est le cas de la forme proposée. Pour plus d'informations sur les propriétés de ce type de construction, le lecteur peut se référer à [25, p. 141].

Le corollaire 3.5 détaille la construction d'un préconditionneur LMP empilé de façon multiplicative. En effet, lorsque deux préconditionneurs LMP sont empilés, tous les vecteurs utilisés dans les constructions sont conjugués par rapport à la matrice à préconditionner pour obtenir un cluster du spectre autour de 1 le plus grand possible.

Corollaire 3.5. [21, p.48]
Soit A, une matrice d'ordre n symétrique définie positive. Prenons deux ensembles de vecteurs linéairement indépendants

$$S^{(1)} = \{ s_1^{(1)}, \ldots, s_{l_1}^{(1)} \} \text{ et } S^{(2)} = \{ s_1^{(2)}, \ldots, s_{l_2}^{(2)} \}$$

avec $l_1 + l_2 \leq n$, et supposons que

1. *K_{l_1} soit construit sur base de la formule* (3.1) *avec $S^{(1)}$ et A la matrice à préconditionner,*
2. *K_{l_2} soit construit sur base de la formule* (3.1) *avec $S^{(2)}$ et $M = G_{l_1}^{(1)^T} A G_{l_1}^{(1)}$ la matrice à préconditionner telle que $G_{l_1}^{(1)} G_{l_1}^{(1)^T} = K_{l_1}$.*

Alors K_{l_2} peut s'écrire

$$\begin{aligned} K_{l_2} = \ & [I_n - S^{(2)} (S^{(2)^T} M S^{(2)})^{-1} S^{(2)^T} M] \, [I_n - M S^{(2)} (S^{(2)^T} M S^{(2)})^{-1} S^{(2)^T}] \\ & + S^{(2)} (S^{(2)^T} M S^{(2)})^{-1} S^{(2)^T} \end{aligned}$$

et sa forme factorisée est $K_{l_2} = G_{l_2}^{(2)} G_{l_2}^{(2)^T}$.

De plus, si l'hypothèse

$$S^{(1)^T} A\, S^{(2)} = 0$$

est vérifiée, alors

$$G_{l_2}^{(2)^T} G_{l_1}^{(1)^T} A\, G_{l_1}^{(1)} G_{l_2}^{(2)}$$

a un cluster de valeurs propres à 1 de multiplicité $m \geq l_1 + l_2$.

3.2 Formulation avec des vecteurs A-conjugués

Dans la suite, nous considérons que la matrice M correspond à la matrice identité. La construction d'un préconditionneur LMP (3.1) exige que les colonnes de la matrice $S = [s_1, \ldots, s_l]$ soient linéairement indépendantes. Lorsque celles-ci vérifient la propriété d'A-conjugaison, l'écriture de (3.1) peut être simplifiée en évitant le calcul de certains produits matrices-vecteurs.

Ainsi, les vecteurs $s_1, \ldots, s_l$ vérifie la propriété

$$s_j^T A\, s_i \begin{cases} = 0 & \text{pour } i \neq j, \\ > 0 & \text{pour } j = i, \end{cases}$$

pour $i, j = 1, \ldots, l$, qui signifie que les vecteurs sont A-conjugués. La forme générale d'un préconditionneur LMP (3.1) avec $M = I_n$ est

$$K_l = [\, I_n - S(S^T A\, S)^{-1} S^T A\,]\, [\, I_n - A\, S (S^T A\, S)^{-1} S^T\,] + S (S^T A\, S)^{-1} S^T. \tag{3.4}$$

Nous remarquons la présence du calcul du produit $S^T AS$ qui peut être simplifié si les colonnes de S sont conjuguées par rapport à la matrice A. En effet, le calcul des produits $s_j^T A\, s_i$ pour $i \neq j$ devient

$$S (S^T A\, S)^{-1} S = \sum_{i=1}^{l} \frac{s_i s_i^T}{s_i^T A\, s_i}.$$

Sous cette hypothèse, la forme générale (3.4) peut alors s'écrire

$$K_l = \left(I_n - \sum_{i=1}^{l} \frac{s_i s_i^T}{s_i^T A\, s_i} A \right) \left(I_n - \sum_{i=1}^{l} A\, \frac{s_i s_i^T}{s_i^T A\, s_i} \right) + \sum_{i=1}^{l} \frac{s_i s_i^T}{s_i^T A\, s_i} \tag{3.5}$$

Les produits As_i sont des quantités utilisées par l'algorithme du gradient conjugué. Il n'est pas nécessaire de les calculer et leur application ne requiert que quelques opérations vectorielles. Nous pouvons donc considérer que l'application de K_l est peu coûteuse lorsque l est relativement petit.

De plus, la conjugaison par rapport à la matrice A des vecteurs S de la construction (3.5) permet de définir une formulation de récurrence comme indiqué dans la proposition 3.6.

Proposition 3.6. [21, p. 70]
Soit A, une matrice symétrique définie positive d'ordre n et $S = [s_1, \ldots, s_l] \in \mathbb{R}^{n \times l}$. *Supposons que* $K_0 = I_n$. *Si les colonnes de la matrice S sont linéairement indépendantes et conjuguées par rapport à la matrice A, alors,*

$$K_l = \left(I_n - \frac{s_l s_l^T}{s_l^T A s_l} A \right) K_{l-1} \left(I_n - A \frac{s_l s_l^T}{s_l^T A s_l} \right) + \frac{s_l s_l^T}{s_l^T A s_l}. \tag{3.6}$$

Dans la suite, la seule propriété que nous imposerons est la A-conjugaison des vecteurs s_i, en plus du fait qu'ils soient linéairement indépendants. Ces propriétés sont garanties pour certains ensembles de vecteurs comme les vecteurs propres, les vecteurs de Ritz et les directions de descente générées par des méthodes dites de Krylov. Ces trois ensembles particuliers donnent lieu à trois instances spécifiques de la classe des LMP.

3.3 Instances de la classe des LMP

Dans cette section, nous allons explorer trois instances concrètes de la classe des LMP basées sur des vecteurs spécifiques sans utiliser de préconditionneur de premier niveau. Nous mentionnerons comment les vecteurs qui permettent la construction ont été obtenus, et les propriétés qu'ils vérifient pour aboutir à l'expression du *spectral-LMP*, *Ritz-LMP* et *quasi-Newton-LMP*. Les détails de toutes les constructions sont disponibles dans la thèse de J. Tshimanga [21, p. 75-84].

Pour construire un préconditionneur, l'idéal est de disposer facilement d'informations. Dans la suite, l'information contenue dans les vecteurs permettant la construction du préconditionneur sera collectée grâce à l'algorithme du gradient conjugué non préconditionné sur un système utilisant la matrice A. Les préconditionneurs ainsi créés pourront être utilisés dans la résolution de suites de systèmes dont la matrice A est identique mais dont le membre de droite change. Il s'agit des systèmes à seconds membres multiples.

3.3.1 Le spectral-LMP

Le premier préconditionneur proposé utilise l'information spectrale de la matrice A d'ordre n symétrique définie positive. Nous travaillons avec les paires propres (λ_i, v_i) avec $i = 1, \ldots, l$ telles que

$$A v_i = \lambda_i v_i, \tag{3.7}$$

et qui vérifient pour $i, j = 1, \ldots, l$ la propriété d'orthogonalité

$$v_j^T v_i = \begin{cases} 0 & \text{pour } i \neq j, \\ 1 & \text{pour } i = j, \end{cases}$$

et la propriété de conjugaison par rapport à la matrice A

$$v_j^T A v_i = \begin{cases} 0 & \text{pour } i \neq j, \\ \lambda_i > 0 & \text{pour } i = j. \end{cases}$$

Proposition 3.7. [21, p. 75]
Le préconditionneur ***spectral-LMP*** *construit sur base de* (3.4) *à partir des paires spectrales* (λ_i, v_i) *de A avec* $i = 1,\ldots,l$, *s'écrit*

$$K_l = \prod_{i=1}^{l} \left[I_n - \left(1 - \frac{1}{\lambda_i}\right) v_i v_i^T \right], \tag{3.8}$$

et sa forme factorisée est

$$G_l = \prod_{i=1}^{l} \left[I_n - \left(1 - \frac{1}{\sqrt{\lambda_i}}\right) v_i v_i^T \right], \tag{3.9}$$

avec $K_l = G_l G_l^T$.

Nous observons que la construction ne nécessite aucun inversion de matrice et que la matrice A n'est pas utilisée.

3.3.2 Le Ritz-LMP

Le préconditionneur *Ritz-LMP* a été proprosé pour la première fois par J. Tshimanga et ses co-auteurs, et se construit à partir de l'information de Ritz. Les paires de Ritz (θ_i, z_i) avec $i = 1\ldots,l$ vérifient la propriété d'orthogonalité

$$z_j^T z_i = \begin{cases} 0 & \text{pour } i \neq j, \\ 1 & \text{pour } i = j, \end{cases}$$

la propriété de conjugaison par rapport à A

$$z_j^T A z_i = \begin{cases} 0 & \text{pour } i \neq j, \\ \theta_i > 0 & \text{pour } i = j, \end{cases}$$

et

$$z_i^T q_{l+1} = 0,$$
$$q_{l+1}^T q_{l+1} = 1,$$

ainsi que

$$A z_i = \theta_i z_i + e_l^T y_i \eta_l q_{l+1} \tag{3.10}$$

pour $i = 1,\ldots,l$, avec e_l la l^e colonne de la matrice identité d'ordre l et η_i, y_i et q_{l+1} respectivement un scalaire et des vecteurs définis par l'algorithme de Lanczos 2.3.

Proposition 3.8. [21, p. 78]
Le préconditionneur ***Ritz-LMP*** *construit sur base de* (3.4) *à partir des informations de Ritz* (θ,v_i) *obtenues par l'algorithme de Lanczos, s'écrit*

$$K_l = \prod_{i=1}^{l} \left[I_n - \left(1 - \frac{1}{\theta_i} - \omega_i^2\right) z_i z_i^T - \omega_i (z_i q_{l+1}^T + q_{l+1} z_i^T) \right], \tag{3.11}$$

et sa forme factorisée est

$$G_l = \prod_{i=1}^{l} \left[I_n - \left(1 - \frac{1}{\sqrt{\theta_i}}\right) z_i z_i^T - \omega_i z_i q_{l+1}^T \right], \tag{3.12}$$

avec

$$\omega_i = \frac{e_l^T\, y_i\, \eta_l}{\theta_i}, \tag{3.13}$$

pour $i = 1, \ldots, l$ *et telle que* $K_l = G_l\, G_l^T$.

La proposition 3.8 donne l'expression du Ritz-LMP construit avec un vecteur supplémentaire q_{l+1} qui n'est pas un vecteur de Ritz. Ce vecteur provient de l'algorithme de Lanczos effectué sur A et permet de construire les expressions de ce préconditionneur plus facilement. La ressemblance du Ritz-LMP avec le spectral-LMP sera discutée plus loin.

3.3.3 Le quasi-Newton-LMP

Cette instance est loin d'être récente puisqu'elle fut proposée par J. Nocedal et J. L. Morales [31] début des années 2000. L'idée est de mettre à jour la direction conjuguée en utilisant la formule de quasi-Newton proposée par Powel en 1957. Ainsi, ils ont introduit un préconditionneur *L-BFGS* se basant sur la formule BFGS et utilsant une petite mémoire. Le préconditionneur quasi-Newton-LMP que nous allons utiliser est similaire et découle de la formulation (3.1).

Lors de l'exécution de l'algorithme du gradient conjugué non préconditionné pour solutionner le système linéaire avec la matrice A, nous allons collecter des directions conjuguées p_i qui vérifient les propriétés suivantes

$$\begin{aligned} p_l^T\, A\, p_i &= 0, \\ r_l^T\, p_i &= 0, \\ r_l^T\, r_i &= 0, \end{aligned}$$

pour $i = 1, \ldots, l-1$.

Proposition 3.9. [21, p. 82]
Le préconditionneur ***quasi-Newton LMP*** *construit sur base de* (3.5) *à partir des paires* $(p_i,\, Ap_i)$ *avec* p_i *les directions de descente générées par l'algorithme du gradient conjugué, s'écrit*

$$K_l = \left(I_n - \frac{p_l\, p_l^T}{p_l^T\, A\, p_l} A \right) K_{l-1} \left(I_n - A\, \frac{p_l\, p_l^T}{p_l^T\, A\, p_l} \right) + \frac{p_l\, p_l^T}{p_l^T\, A\, p_l}, \tag{3.14}$$

avec $K_0 = I_n$. *De plus en utilisant les résidus, la forme factorisée est*

$$G_l = \left[I_n - p_l \left(\frac{p_l^T\, A}{p_l^T\, A\, p_l} + \frac{1}{\sqrt{p_l^T\, A\, p_l}}\, \frac{r_l^T}{\|r_l\|_2} \right) \right] G_{l-1}, \tag{3.15}$$

avec $K_l = G_l\, G_l^T$.

La forme (3.14) est construite de manière récursive à partir de la formule (3.6). Afin d'être efficace, il est préférable de collecter les vecteurs Ap_i afin de ne pas devoir effectuer ce produit lors de la construction du préconditionneur. Autrement dit, à chaque itération du gradient conjugué, la paire (p_i, Ap_i) doit être stockée. De plus, la forme factorisée (3.15) requiert le stockage de la norme du résidu de l'algorithme du gradient conjugué.

Remarque [21, p. 84] : Il est possible d'imaginer que nous puissions utiliser d'autres informations que les directions conjuguées, comme les résidus du système linéaire ou les vecteurs de Lanczos pour construire un LMP. Cependant, ces vecteurs sont orthogonaux mais ils ne possèdent pas la propriété de A-conjugaison.

3.3.4 Quelques comparaisons

Nous allons comparer brièvement le Ritz-LMP aux deux autres instances de la classe des LMP présentées et commenter leurs similarités : d'une part, le quasi-Newton-LMP, qui utilise une base du sous-espace de Krylov, et d'autre part, le spectral-LMP qui est basé sur de l'information spectrale.

Ritz-LMP et quasi-Newton-LMP

L'information permettant de construire les préconditionneurs Ritz-LMP et quasi-Newton-LMP est collectée lors du gradient conjugué non préconditionné. Ainsi, les directions de descente et les vecteurs de Ritz collectés forment respectivement une base du sous-espace de Krylov. Le théorème 3.10 qui souligne l'équivalence entre les deux versions de LMP, à condition que toute l'information disponible provenant du gradient conjugué soit utilisée.

Théorème 3.10. [21, p. 90]
Les préconditionneurs Ritz-LMP et quasi-Newton-LMP sont analytiquement équivalents si leur construction utilise toute l'information disponible (les directions de descente ou les vecteurs de Ritz) collectée lors d'une procédure de type GC appliquée à une même matrice.

En ce qui concerne le stockage en mémoire, le quasi-Newton-LMP utilise l paires $(p_i, A\,p_i)$ donc un total de $2l$ vecteurs. De son coté, le Ritz-LMP utilise $l+1$ vecteurs ce qui correspond à environ la moitié du quasi-Newton-LMP.

Spectral-LMP et Ritz-LMP

Le spectral-LMP et le Ritz-LMP utilisent l'information spectrale relative à la matrice à préconditionner. Théoriquement, le premier utilise les vecteurs propres et valeurs propres exacts alors que le second se base sur leurs approximations. Il est bien entendu certain que si l'information exacte est disponible, le spectral-LMP est à privilégier, puisqu'il va permettre la clusterisation autour de 1 des valeurs propres connues. A l'inverse, lorsqu'elles sont inconnues, le calcul de ces informations est coûteux ce qui justifie l'approche proposée par J. Tshimanga et ses co-auteurs.

Lors de nos expérimentations, le spectral-LMP va être construit à partir d'information spectrale inexacte. En effet, nous allons utiliser l'information de Ritz pour approximer les paires propres. J. Tshimanga a montré que le spectral-LMP construit à partir de l'information de Ritz se comporte comme le Ritz-LMP à condition que les paires de Ritz aient suffisamment convergé vers les paires propres.

La quantité d'informations en mémoire permettant la construction de ces deux préconditionneurs est quasi identique. En effet, le Ritz-LMP nécessite un vecteur supplémentaire q_{l+1} comparé au spectral-LMP.

3.3.5 Application des instances

Nous allons utiliser les trois instances de la classe des LMP créées dans le contexte de résolution d'une séquence de systèmes linéaires à second membres multiples, $Ax = b^{(i)}$, $i = 1, 2, \ldots$ par le gradient conjugué. Nous allons construire un LMP pour le deuxième système linéaire $Ax = b^{(2)}$ à partir de l'information récoltée lors de la résolution du premier $Ax = b^{(1)}$. L'objectif est de comparer les performances des trois instances LMP que nous avons introduites selon le nombre de colonnes dans S.

Nous allons procéder de la manière suivante. L'algorithme du gradient va être appliqué avec un nombre d'itérations l fixé pour résoudre le premier système $Ax = b^{(1)}$ dont le membre de droite est généré aléatoirement. Pendant cette

résolution, l vecteurs (directions conjuguées ou vecteurs de Ritz) nécessaires à la construction de K_l sont collectés. Le second système possède la même matrice A et un autre membre de droite $b^{(2)}$ généré lui aussi aléatoirement. Ensuite, après avoir construit le préconditionneur K_l, le second système est résolu par l'algorithme du gradient conjugué préconditionné. Rappelons que pour le spectral-LMP, les paires propres sont en pratique les paires de Ritz.

Les performances des trois instances vont être mesurées à partir de la norme A de l'erreur sur la solution

$$\|x_k - x^*\|_A = \sqrt{(x_k - x^*)^T A (x_k - x^*)} \tag{3.16}$$

à chaque itération. En effet, nous disposons de la solution exacte x^* car nous la fixons en générant aléatoirement pour ensuite calcul $b^{(2)} = Ax^*$ afin d'obtenir le système à résoudre. Pour chaque préconditionneur, nous allons déterminer le nombre d'itérations requis par le gradient conjugué préconditionné pour que (3.16) soit inférieure à une tolérance fixée à 10^{-5}.

A des fins d'illustration, nous avons choisi d'utiliser les matrices 1 et 3 présentées à la section 3.1. Notons que les deux autres matrices donnent des résultats similaires à ceux que nous obtenons pour ces deux matrices. Tout d'abord, montrons que l'utilisation des trois instances LMP est pertinente pour $l = n/4$. Les figures 3.7 et 3.8 illustrent l'amélioration de la convergence lors du préconditonnement LMP. Dans les deux résolutions, les courbes correspondant au quasi-Newton-LMP et au Ritz-LMP se superposent. La résolution du système associé à la matrice 1 passe de 71 itérations sans préconditionnement à 44 itérations quel que soit le LMP utilisé. Quant au système associé à la matrice 3, sans préconditionnement 293 itérations sont nécessaires pour que la norme A de l'erreur soit inférieure au seuil fixé. En utilisant le quasi-Newton-LMP ou le Ritz-LMP, ce nombre est réduit à 141. Le résultat est moins satisfaisant avec le spectral-LMP puisqu'il fait chuter le nombre d'itérations à 169.

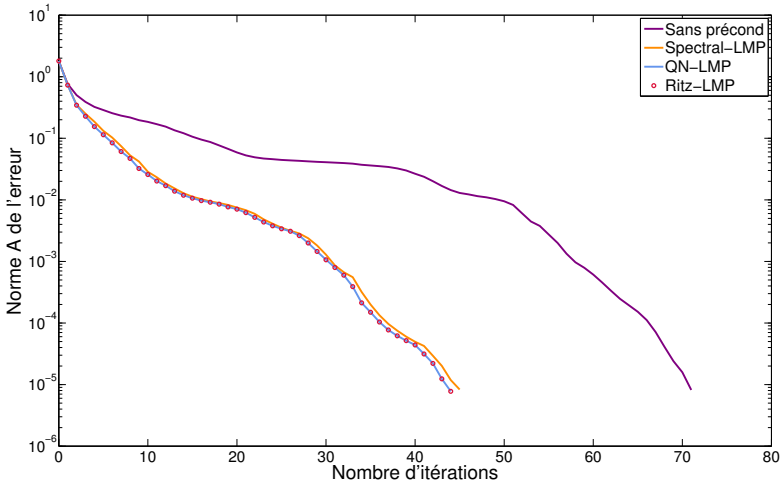


FIGURE 3.7 – Résolution pour la matrice 1.

Intéressons-nous à la performance des LMP selon le nombre de vecteurs utilisés pour leur construction. Pour chaque système, la figure de droite de 3.9 et 3.10 compare les performances de chaque préconditionneur pour un nombre de vecteurs dans S. Pour ce même nombre de vecteurs, la figure de gauche indique la borne maximale de l'erreur de l'estimation des paires propres par les paires de Ritz (θ_i, z_i), c'est-à-dire la *Max Ritz Error Bound*. Ainsi pour chaque ensemble S de taille l_S, nous traçons $\max_{i \in l_S} \|Az_i - \theta_i z_i\|_2$.

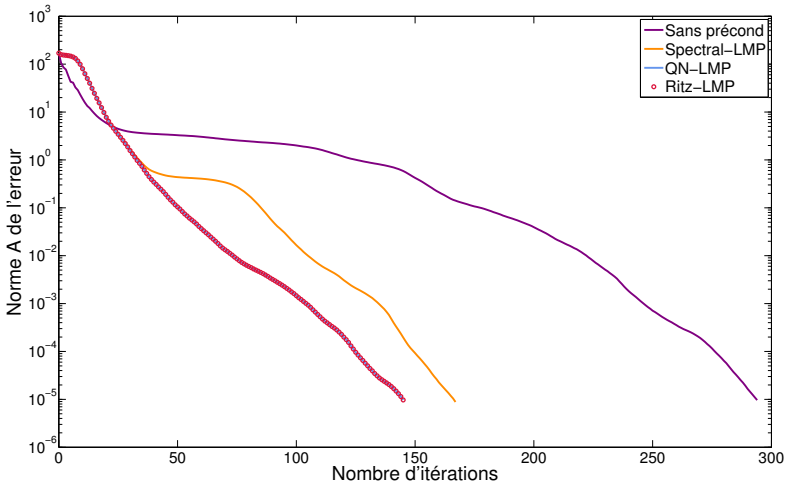


FIGURE 3.8 – Résolution pour la matrice 3.

Nous observons pour les trois instances que la performance augmente avec le nombre de vecteurs. En effet, le nombre d'itérations diminue avec le nombre de vecteurs ce qui est cohérent avec les propriétés générales des LMP concernant la clusterisation du spectre. L'équivalence entre les préconditionneurs quasi-Newton-LMP et Ritz-LMP mentionnée dans le théorème 3.10 est vérifiée puisque leurs courbes se superposent. Quant au spectral-LMP, il donne de moins bons résultats que les deux autres instances. Cependant, si la borne de l'erreur de l'estimation des paires propres exactes par les paires de Ritz est petite, le spectral-LMP utilisant les paires de Ritz a un comportement proche de celui du Ritz-LMP.

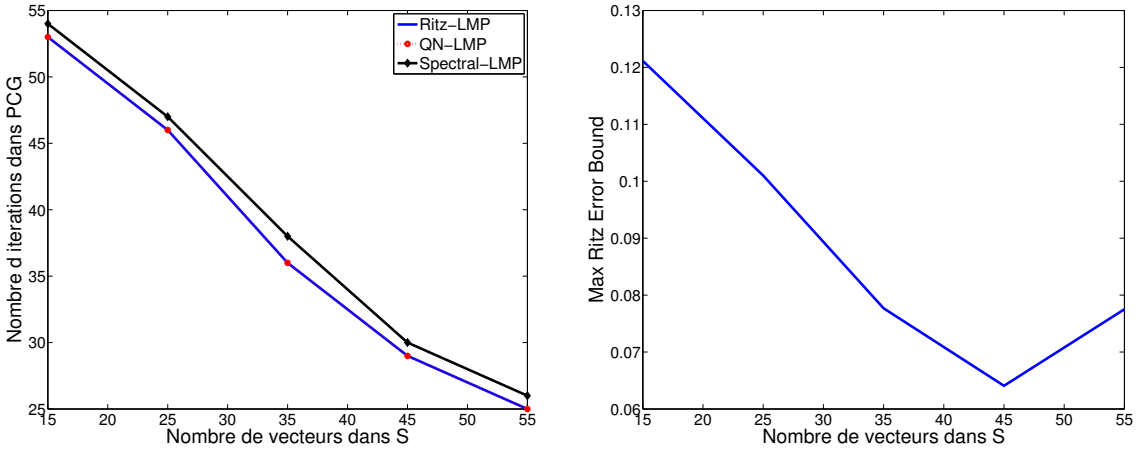


FIGURE 3.9 – Performances des trois instances en fonction du nombre de vecteurs dans S pour la matrice 1.

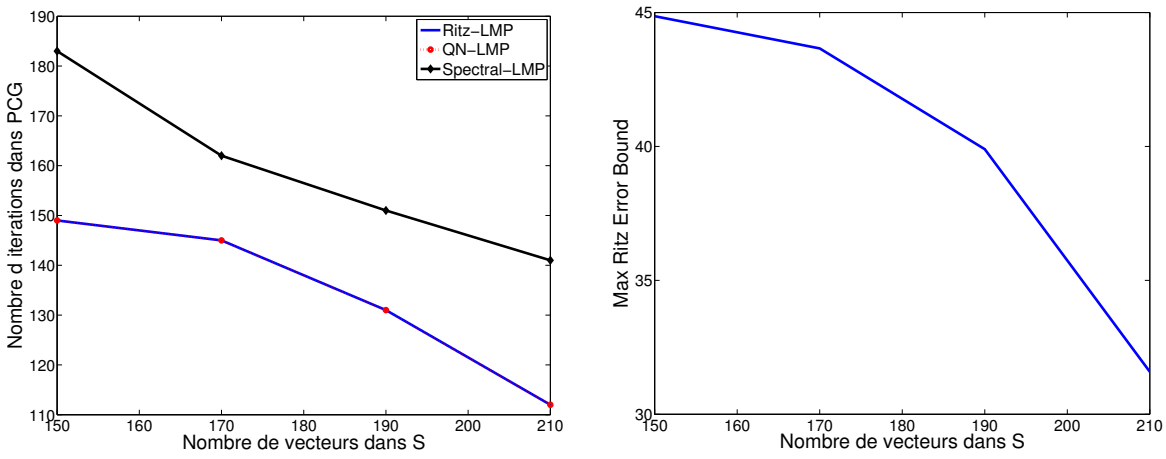


FIGURE 3.10 – Performances des trois instances en fonction du nombre de vecteurs dans S pour la matrice 3.

3.4 Préconditionnement avec perturbation

Cette section met en avant le comportement du préconditionneur K_l lorsque la matrice à préconditionner A est légèrement perturbée pour devenir $\tilde{A}$ avant son préconditionnement. Nous allons observer l'effet du préconditionneur

$$K_l = \left(I_n - \sum_{i=1}^{l} \frac{s_i s_i^T}{s_i^T A s_i} A\right)\left(I_n - \sum_{i=1}^{l} A \frac{s_i s_i^T}{s_i^T A s_i}\right) + \sum_{i=1}^{l} \frac{s_i s_i^T}{s_i^T A s_i} \tag{3.17}$$

sur la matrice $\tilde{A}$.

Dans sa thèse, J. Tshimanga [21, p. 104-108] présente différents résultats qui nous permettent d'affirmer que le cluster autour de 1 de la matrice $K_l \tilde{A}$ peut aussi être observé lorsque A ne change pas significativement, ou que les directions conjuguées par rapport à la matrice A restent suffisamment conjuguées par rapport à la matrice perturbée $\tilde{A}$.

Nous retiendrons simplement que si nous disposons des vecteurs $s_1, \ldots, s_l$ conjugués par rapport à la matrice A et ν est un nombre réel tel que $A + \nu A$ est toujours définie positive, alors, les vecteurs A-conjugués sont aussi conjugués à la matrice $A + \nu A = (1+\nu)A$. Ainsi, le LMP (3.17) peut être construit en utilisant ces vecteurs. En théorie, la perturbation ne détruit pas le cluster de valeurs propres autour de 1 obtenu en préconditionnant.

Tests numériques

A présent, illustrons ce que nous venons d'expliquer à l'aide de deux systèmes dont la matrice varie de l'un à l'autre. Les tests suivants sont fondés sur le préconditionneur quasi-Newton-LMP construit à partir des directions conjuguées $s_1, \ldots, s_l$ obtenues lors de l'application de l'algorithme du gradient conjugué non préconditionné au système $Ax = b$. La matrice perturbée $\tilde{A}$ est construite en ajoutant à A une matrice symétrique définie positive générée aléatoirement, ayant les mêmes éléments non nuls que A. Cette perturbation est calibrée par une fraction de la norme de la matrice du système $\tau \|A\|_2$, avec τ une valeur réelle relativement petite que nous choisirons comme 10^{-3} ou 10^{-5}.

L'objectif est d'illustrer l'effet du préconditionneur quasi-Newton-LMP construit sur base de vecteurs $s_1, \ldots, s_l$ non conjugués par rapport à la matrice $\tilde{A}$ du système à résoudre. L'algorithme 3.1 permet de conjuguer ces vecteurs par rapport à la matrice $\tilde{A}$ en s'inspirant du processus d'orthogonalisation de Gram-Schmidt [24, p. 254] en remplaçant le produit euclidien par le produit scalaire incluant la matrice $\tilde{A}$. Cet algorithme sera utilisé pour reconjuguer les vecteurs avant la construction de K_l par la suite.

Dans un premier temps, la figure 3.11 montre qu'il est pertinent d'utiliser le préconditionneur pour résoudre le système perturbé. Le test est réalisé avec la matrice 1 perturbée avec un facteur 10^{-3}. Notons que les 3 autres matrices proposées à la section 3.1 donnent des résultats similaires pour les facteurs 10^{-3} et 10^{-5}. La construction du préconditionneur se fait avec la matrice A et $l = n/4$ vecteurs, c'est-à-dire 25 vecteurs. La figure représente le nombre d'itérations nécessaire pour que le résidu relatif soit inférieur à 10^{-4}. Lors de la résolution du système initial, le GC et le GCP requièrent respectivement 61 et 31 itérations. Tandis que pour le système perturbé, bien qu'il converge plus rapidement (53 itérations) que le système initial, la version préconditionnée donne de moins bons résultats (36 itérations). En effet, pour le système initial, l'amélioration est de 30 itérations contre 17 pour le système perturbé.

Algorithme 3.1 – Procédure de conjugaison par rapport à $\tilde{A}$

Soit $s_1, \ldots, s_l$ et $\tilde{A}$ donnés

for $i = 1 : l$

  for $j = i+1 : l$

$$s_i \longleftarrow s_i - \frac{s_j^T \tilde{A} s_i}{s_j^T \tilde{A} s_j} s_j$$

  end(for)

$$z_i \longleftarrow \frac{z_i}{\sqrt{s_i^T \tilde{A} s_j}}$$

end(for)

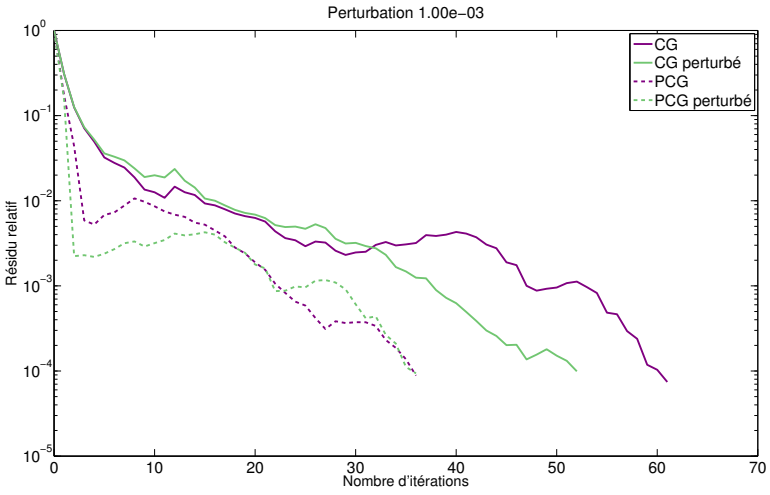


FIGURE 3.11 – Résolutions du système associé à la matrice 1 en mauve et à la matrice 1 avec une perturbation de 10^{-3} en vert.

Afin d'analyser l'impact de la reconjugaison des vecteurs sur la résolution préconditionnée, nous allons maintenant comparer graphiquement l'effet du préconditionnement de $\tilde{A}$ par

- K_l construit à partir de la matrice A et des vecteurs de S conjugués par rapport à A,
- K_l construit à partir de la matrice $\tilde{A}$ et des vecteurs de S reconjugués par rapport à $\tilde{A}$,

et ce, en fonction du nombre de vecteurs dans S. Remarquons que contrairement à ce que nous avions présenté au début de ce chapitre, la seconde forme est construite sur base de la matrice à préconditionner et non plus celle qui a permis de collecter l'information.

Une nouvelle fois, nous allons illustrer le nombre d'itérations nécessaire pour que le résidu relatif soit inférieur à 10^{-4} et ce pour deux perturbations, 10^{-3} et 10^{-5}. Aux figures 3.12 et 3.13, l'impact de la reconjugaison des vecteurs par rapport à la matrice $\tilde{A}$ est visible. Pour la plus petite pertubation, 10^{-5}, nous observons que le préconditionneur avec reconjugaison, en ligne pointillée, permet de gagner une itération et ce pour l fixé à 15 et 25. Pour la perturbation de 10^{-3}, le comportement est similaire et l'écart est de deux itérations.

Précédemment, nous avons observé que le nombre de vecteurs dans S a un impact sur la performance du LMP. Il est donc opportun d'analyser la performance du LMP, avec et sans reconjugaison des vecteurs, selon le nombre d'itérations utilisées dans le GCP afin que la $\tilde{A}$-norme de l'erreur sur la solution atteigne une tolérance fixée à 10^{-5}.

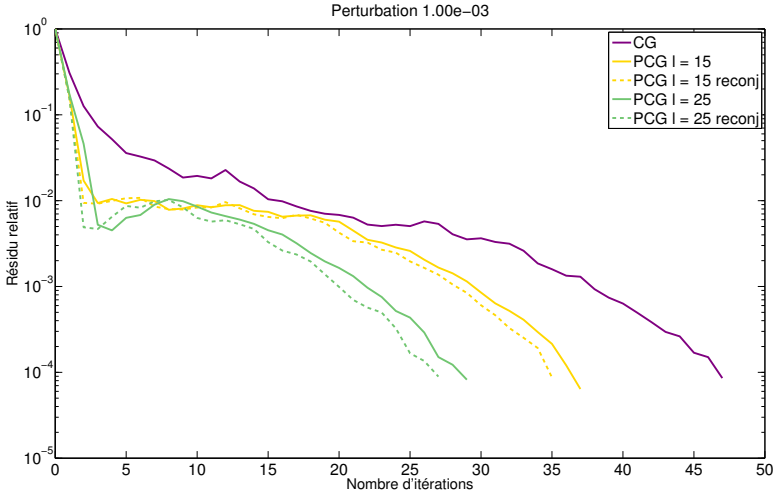


FIGURE 3.12 – Résolution du système associé à la matrice 1 avec une perturbation de 10^{-3}, et ce, avec et sans reconjuguaison des vecteurs de S.

En observant la figure 3.14, nous remarquons que les systèmes perturbés utilisant une procédure de reconjugaison des vecteurs de S donnent de meilleurs résultats que ceux sans reconjugaison. Cependant, le gain est très faible puisqu'il correspond à une ou deux itérations. Lors de la résolution de systèmes perturbés et de l'utilisation de LMP, il peut être opportun de reconjuguer par rapport à la matrice à préconditionner les vecteurs contenant l'information pour la construction. Cependant, il faut garder à l'esprit que le coût de cette opération doit être abordable et justifié par le gain apporté par la suite.

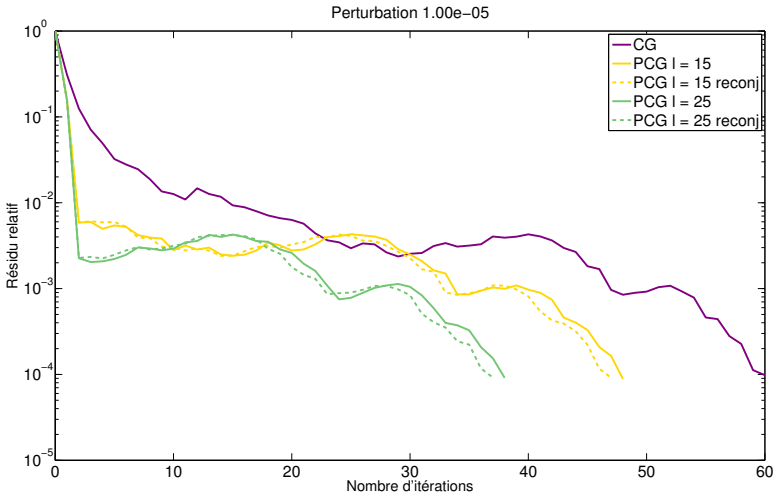


FIGURE 3.13 – Résolution du système associé à la matrice 1 avec une perturbation de 10^{-5}, et ce, avec et sans reconjuguaison des vecteurs de S.

Pour conclure, nous avons supposé que la matrice A n'est pas significativement différente de $\tilde{A}$. De ce fait, les vecteurs A-conjugués peuvent être utilisés pour préconditionner le système perturbé. Dans la littérature, il n'est pas spécifié jusqu'à quel point le LMP reste efficace dans cette situation, comme le précise l'article [27]. En effet, nous disposons uniquement d'analyses empiriques pour lesquelles les matrices ne varient pas assez que pour atténuer l'effet positif du LMP.

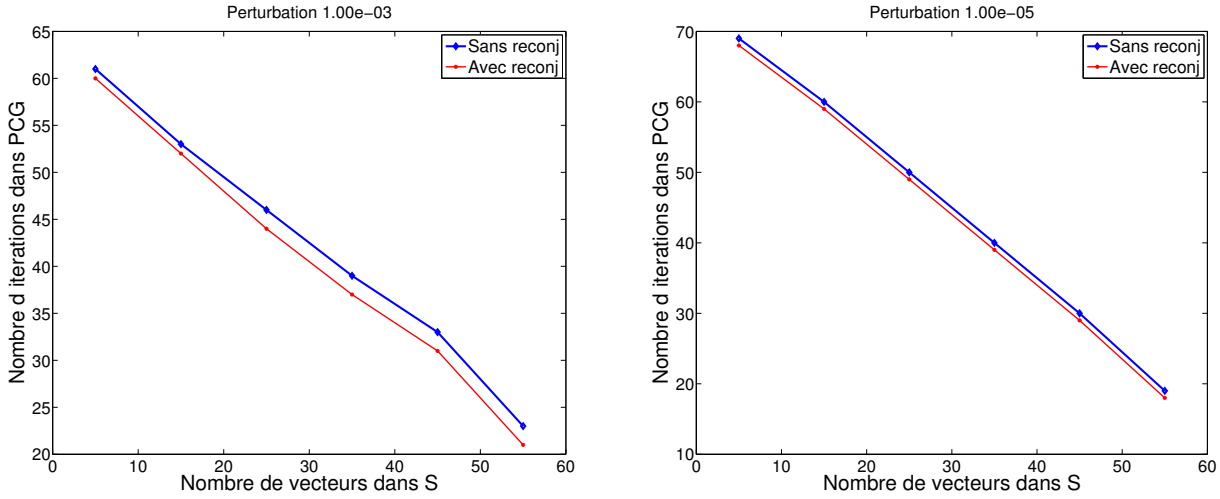


FIGURE 3.14 – Performances du quasi-Newton-LMP, avec et sans reconjugaison, en fonction du nombre de vecteurs dans S pour la matrice 1 associée à une perturbation de 10^{-3} à gauche et 10^{-5} à droite.

3.5 Conclusion

Ce chapitre nous a permis d'introduire les préconditionneurs LMP qui sont des matrices symétriques définies positives. La formulation générale requiert la matrice à préconditionner, ainsi qu'un ensemble de vecteurs linéairement indépendants. Nous avons abordé les propriétés et résultats les plus importants. Retenons en particulier que, d'un point de vue spectral, l'application du préconditionneur ne détruit pas, s'il existe préalablement, le cluster autour de 1 des valeurs propres, et il n'y a pas d'expansion du spectre.

Le spectral-LMP, le Ritz-LMP et le quasi-Newton-LMP ont été présentés plus en détails. Dans un contexte de résolution de systèmes en série, l'attrait pour ces trois préconditionneurs réside dans la collecte des vecteurs A-conjugués. En effet, ceux-ci peuvent être obtenus à un coût relativement faible puisqu'ils sont automatiquement produits par l'algorithme du gradient conjugué. Dans nos expérimentations, le spectral-LMP et le Ritz-LMP utilisent la même information de Ritz bien que leur formulation soit différente.

Nos analyses ont mis en avant que les LMP sont sensibles au nombre de vecteurs utilisés lors de leur construction ainsi qu'à leur qualité. Il n'est pas clairement établi si tous les vecteurs collectés doivent être impliqués. Par exemple, le spectral-LMP donne de bons résultats lorsqu'il est construit à partir de l'information de Ritz, si celle-ci a suffisamment convergé vers les paires propres. Néanmoins, il est plus opportun d'utiliser le Ritz-LMP puisque sa formulation est plus générale. Nous avons observé que Ritz-LMP se comporte comme une version stabilisée du spectral-LMP si l'information de Ritz est utilisée.

Finalement, dans le contexte de résolution de série de systèmes, nous avons fait l'hypothèse que la matrice des systèmes ne varie pas significativement. Ainsi, nous avons analysé le comportement du quasi-Newton-LMP qui, sur l'exemple choisi, donne des résultats probants. Dans le cadre de la résolution du problème des moindres carrés non linéaires par la méthode de Gauss-Newton, il peut être envisageable de reconjuguer les vecteurs. En effet, à chaque itération, un système linéaire symétrique défini positif doit être résolu. Les vecteurs collectés lors des premières résolutions vont permettre de préconditionner les systèmes suivants. Or, entre le premier et le second système, la matrice est généralement différente en raison du processus de linéarisation de la méthode.

Chapitre 4

Application des LMP au recalage d'images

Ce chapitre s'intéresse à la librairie FAIR [1], *Flexible Algorithms for Image Registration*. Tout d'abord, nous introduirons les grandes lignes du processus de recalage d'images médicales dans cette libraire. Nous évoquerons l'approche multi-niveaux ainsi que l'importance du pré-recalage. Ensuite, nous décrirons l'algorithme de Gauss-Newton et nous allons élaborer une stratégie afin d'y introduire les LMP présentés dans le chapitre précédent. Par la suite, cette méthode de résolution sera appliquée à un cas concret de recalage d'images. Nous terminerons ce chapitre par une limitation des LMP dans le cadre de la résolution de systèmes linéaires symétriques définis positifs dans FAIR.

4.1 Le recalage avec FAIR

La librairie d'algorithmes de recalage d'images médicales FAIR est implémentées en Matlab. Elle a été développée par Jan Modersitzki [15] à des fins pédagogiques et académiques. On y trouve entre autres les grandes étapes du recalage comme la discrétisation, la régularisation, l'interpolation et la phase d'optimisation ainsi que leurs implémentations. C'est à partir de ces outils que nous allons pouvoir manipuler le processus de recalage et y intégrer les préconditionneurs LMP.

Dans le processus d'optimisation, FAIR met l'accent sur les méthodes de type Gauss-Newton associées à une recherche linéaire. Les systèmes linéaires sont résolus par des méthodes itératives comme le gradient conjugué et sa version préconditionnée intégrées dans Matlab.

4.1.1 L'approche multi-niveaux

L'approche multi-niveaux (*multilevel*) est recommandée dans FAIR. Elle est très utile pour les images de haute résolution pour lesquelles un recalage est fastidieux lorsqu'il doit être effectué à un niveau élevé c'est-à-dire avec une discrétisation fine. Supposons que nous disposons de deux images discrétisées sur deux grilles centrées comparables dans un même domaine. Les images sont discrétisées en un certain nombre de cellules de même taille qui dépend du niveau de discrétisation souhaité. Par exemple, pour le niveau 3, la grille est composée de $2^3 \times 2^3$ cellules. Au plus le niveau augmente, au plus la discrétisation devient fine. Au lieu de résoudre un problème d'optimisation pour une discrétisation fixée, l'approche multi-niveaux résout une série de problème de recalage en passant d'un niveau à l'autre.

Dans un premier temps, le recalage se fait sur des données à un niveau grossier en fusionnant les cellules de la grille et en leur donnant comme intensité la moyenne de celles des anciennes cellules, réduisant ainsi la taille du

problème. Suite à la résolution du recalage au premier niveau, la transformation obtenue devient le point de départ du recalage à un niveau supérieur et ainsi de suite. La figure 4.1 illustre la stratégie multi-niveaux, les flèches bleues représentent l'utilisation de la solution dans le recalage au niveau supérieur. L'utilisation de la transformation obtenue comme point de départ pour le système suivant permet de converger rapidement.

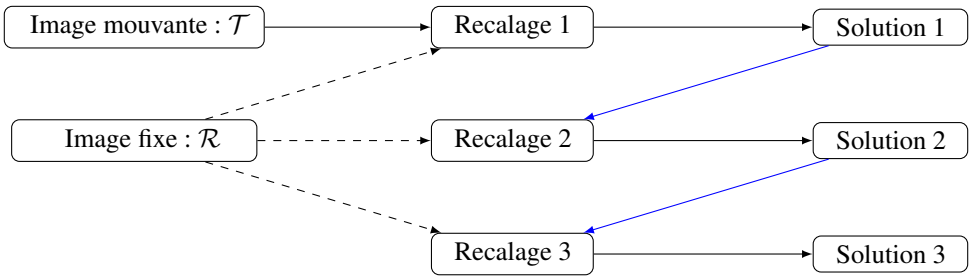
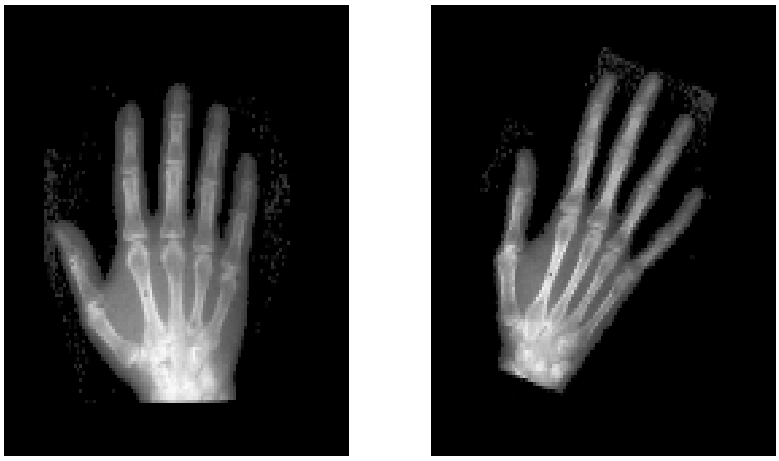


FIGURE 4.1 – Stratégie de résolution multi-niveaux. [32, Modifiée]

Cette approche a pour avantage d'effectuer la quasi-majorité des calculs lors des résolutions à faible niveau. En effet, au plus le niveau augmente, au plus les calculs sont coûteux à effectuer. Dans les niveaux les plus fins, il suffit d'affiner légèrement la transformation pour finalement obtenir un bon résultat pour le niveau souhaité au départ. De plus, les premiers recalages optimisent une fonction objectif plus lisse, ce qui permet de réduire le risque d'être piégé proche d'un minimum local.

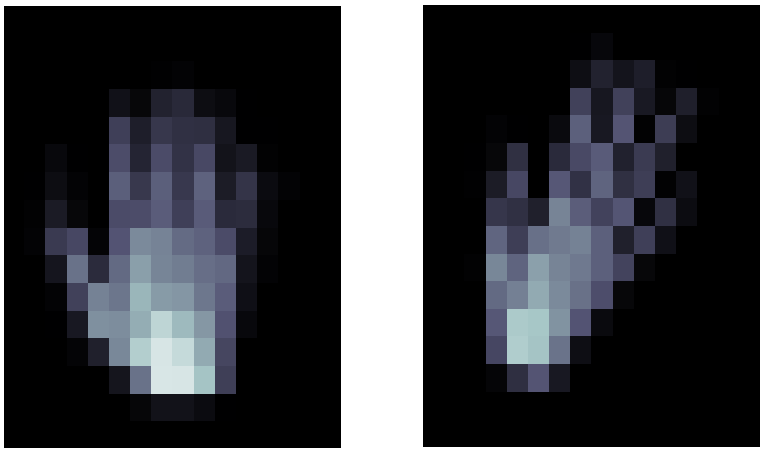
A présent, nous allons nous pencher sur un cas concret de deux images à recaler. Celles-ci sont représentées à la figure 4.2 et sont issues de la librairie des images de test de FAIR [1]. Le processus de recalage sera appliqué à la discrétisation de ces images pour les niveaux 4 et 5 représentés aux figures 4.3 et 4.4.



(a) Image fixe (b) Image mouvante

FIGURE 4.2 – Deux images à recaler. [1]

L'idée est de recaler l'image mouvante sur l'image fixe en lui appliquant une transformation spatiale. Dans un premier temps, nous allons voir la nécessite d'effectuer un pré-recalage. Cette étape consiste en un recalage rigide dont la transformation est composée de rotations et translations ainsi que de transformations affines. Ce premier recalage global va permettre d'améliorer la ressemblance entre les deux images. A partir de l'image mouvante que nous avons recalé une première fois, nous allons de nouveau lui appliquer une transformation mais cette fois non rigide et non paramétrique. Notons que dans FAIR, la transformation est dite non paramétrique car elle n'est plus paramétrisée contrairement au recalage précédent (voir [15] pour plus de détails). Ce second recalage va intervenir globalement et/ou localement en déformant certaines zones de l'image. Dans les illustrations qui vont suivre, le champ de déformation sera de type élastique. Il ne prend pas en compte la composante rigide de la déformation.



(a) Image fixe (b) Image mouvante

FIGURE 4.3 – Images à recaler, niveau 4, $2^4 \times 2^4$ cellules. [1]

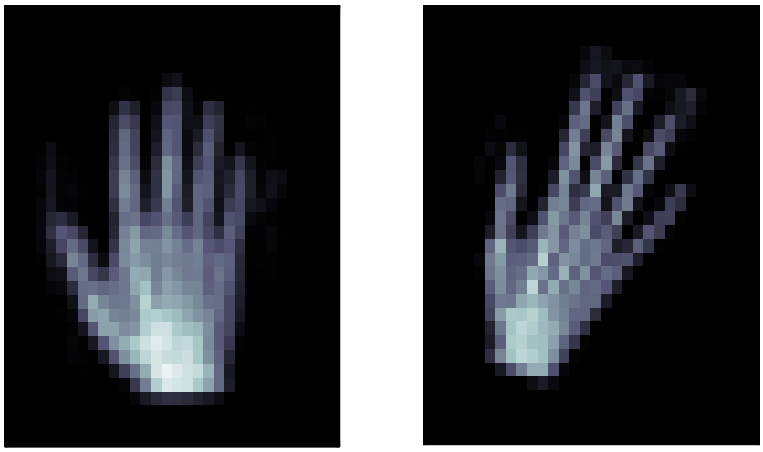


FIGURE 4.4 – Images à recaler, niveau 5, $2^5 \times 2^5$ cellules. [1]

Remarque : Il n'est pas recommandé de résoudre le problème pour un niveau fixé mais bien avec une approche multi-niveaux. Cependant, à des fins d'illustration, nous travaillerons avec des niveaux fixés. Notons que Modersitzki [15] propose une autre approche dite multi-échelles (*multiscale*).

4.1.2 Pré-recalage

Le pré-recalage consiste à recaler une première fois l'image mouvante à l'aide d'une transformation rigide. L'image ainsi obtenue constitue l'image mouvante pour débuter le recalage non rigide. Les résultats de trois recalages à la figure 4.5 permettent de mieux comprendre le rôle important du pré-recalage dans l'application que nous avons choisie. Le résultat obtenu sans pré-recalage n'est pas concluant vu la déformation des doigts. Lorsque l'image mouvante est déformée par une transformation rigide, la ressemblance avec l'image fixe est visible. Cependant, il est possible de déformer localement certaines zones de l'image afin d'améliorer le recalage. La meilleure manière de procéder est de recaler une première fois de manière rigide et d'utiliser l'image mouvante recalée comme point de départ pour un second recalage non rigide cette fois. Notons que la solution du recalage rigide est utilisée comme biais dans le terme de régularisation de la fonction objectif.

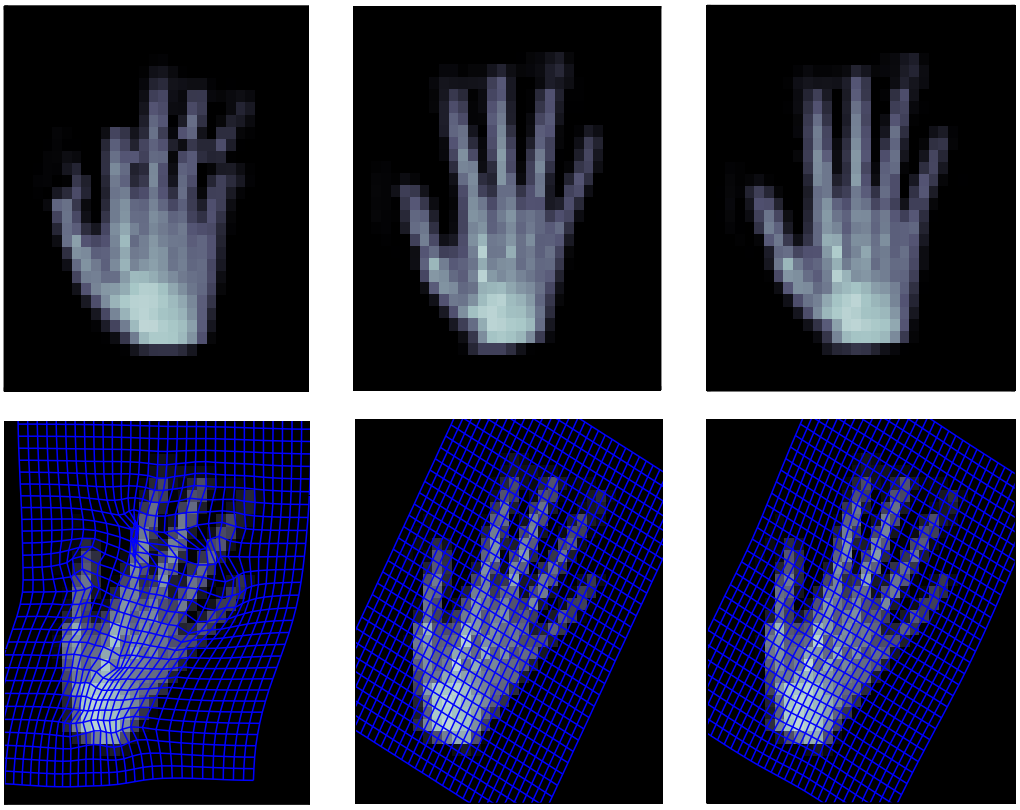


FIGURE 4.5 – En niveau 5, de gauche à droite : résultat après recalage uniquement non rigide, après recalage rigide linéaire, après recalage des deux approches combinées.

4.1.3 Fonctionnement de l'algorithme dans FAIR

Le problème de recalage dans FAIR se résout à l'aide de la méthode de Gauss-Newton. La section 2.3 spécifie que le gradient et l'approximation du Hessien de la fonction objectif doivent être mis à disposition. Nous allons regarder à la forme prise par ces fonctions pour ensuite décrire les principales étapes du recalage telles qu'elles sont implémentées dans FAIR.

Fonctions objectifs et dérivées

Le chapitre 1 a introduit la stratégie d'optimisation et la fonction objectif (1.6) sous la forme discrète qui est, pour rappel,

$$J[y] = \frac{1}{2} \| \mathcal{T}(y) - \mathcal{R} \|_2^2 + \frac{1}{2} \| \mathcal{B}(y - y^{ref}) \|_2^2.$$

Cette fonction peut s'exprimer comme la somme de f et g sachant que

$$f(y) = \frac{1}{2} \| \mathcal{T}(y) - \mathcal{R} \|_2^2 \quad \text{et} \quad g(y) = \frac{1}{2} \| \mathcal{B}(y - y^{ref}) \|_2^2.$$

Ces deux termes vont être traités séparément pour finalement permettre d'obtenir le gradient et le Hessien de la fonction objectif (1.6).

D'une part, le premier terme f correspond au critère de dissimilarité qui mesure la distance entre les deux images. Nous définissons le résidu $r(y) = \mathcal{T}(y) - \mathcal{R}$ qui nous permet d'écrire

$$f(y) = \frac{1}{2} r(y)^T r(y).$$

Ainsi définie, f s'apparente à la forme du problème des moindres carrés (2.6). Ensuite, après avoir défini $Jac(y) = \nabla r(y)$, le gradient est donné par

$$\nabla f(y) = Jac(y)^T \, r(y)$$

et l'approximation du Hessien est donné par

$$\nabla^2 f(y) \approx Jac(y)^T \, Jac(y) \overset{not.}{=} H_f. \tag{4.1}$$

D'autres part, le terme relatif à la régularisation g peut s'écrire sous la forme d'un problème aux moindres carrés

$$g(y) = \frac{1}{2} (y - y^{ref})^T \, \mathcal{B}^T \, \mathcal{B}(y - y^{ref}).$$

Le gradient de g est donné par

$$\nabla g(y) = \mathcal{B}^T \mathcal{B}(y - y^{ref})$$

et le Hessien par

$$\nabla^2 g(y) = \mathcal{B}^T \mathcal{B} \overset{not.}{=} H_g. \tag{4.2}$$

Finalement, nous obtenons le gradient de la fonction objectif donné par

$$\nabla J[y] = Jac(y)^T \, r(y) + \mathcal{B}^T \mathcal{B}(y - y^{ref}) \overset{not.}{=} dJ$$

et le Hessien approximé par

$$\nabla^2 J[y] = H_f + H_g \overset{not.}{=} H. \tag{4.3}$$

Le système d'équations $H_k \, p_k = h_k$ à résoudre à l'étape 2 de l'algorithme 2.1 est de la forme

$$H \, dy = -dJ$$

et permet d'obtenir une direction de descente notée dy .

L'algorithme de résolution

L'algorithme de Gauss-Newton 2.1 tel qu'il a été défini précédemment est adapté dans FAIR. Dans la boucle d'optimisation, il utilise l'algorithme du gradient conjugué 2.2 combiné à une recherche linéaire. Les principales étapes de l'algorithme implémenté dans FAIR sont décrites par l'algorithme 4.1. Notons que la fonction objectif et ses dérivées ont été indicées par c, ce qui signifie qu'elles sont évaluées en y_c, l'itéré courant. Ainsi nous utiliserons les notations J_c, dJ_c et H_c respectivement pour la fonction objectif, le gradient et le Hessien évalués en l'itéré courant.

L'algorithme 4.1 utilise les paramètres par défaut suivants dans son critère d'arrêt :

- $tolJ = 1e-3$: critère d'arrêt sur la fonction objectif ;
- $tolY = 1e-2$: critère d'arrêt sur l'itéré courant ;
- $tolG = 1e-2$: critère d'arrêt sur la norme du gradient.

En ce qui concerne le critère d'arrêt de l'algorithme du gradient conjugué, la tolérance sur la norme du résidu relatif est fixée à 10^{-6}.

Dans la suite, nous parlerons d'itérations externes pour désigner celles de la boucle d'optimisation de l'algorithme 4.1 et d'itérations internes pour désigner celles de la procédure de résolution du système linéaire à l'étape 3. Afin d'accélérer cette résolution, FAIR dispose de méthodes de préconditionnement. Il s'agit essentiellement de la méthode de Gauss-Seidel et de Jacobi (voir Y. Saad [19]). La section suivante traite de l'intégration des préconditionneurs LMP dans la résolution.

Algorithme 4.1 – Gauss-Newton dans le cadre de FAIR pour un niveau fixé

Soit y_c, une transformation initiale donnée par le pré-recalage

Évaluation de J_c, dJ_c et H_c en y_c

$y_{Old} = 0$, $J_{Old} = Jc$, $y_0 = y_c$, $iter = 0$, $arret = 0$ et $maxIter = 20$

while ($arret == 0$)

1. $iter = iter + 1$
2. Vérification du critère d'arrêt :

   $Stop(1) = \mid J_{Old} - J_c \mid \leq tolJ\ (1 + \mid J_c \mid)$

   $Stop(2) = \|y_c - y_{Old}\|_2 \leq tolY\ (1 + \|y_0\|_2)$

   $Stop(3) = \|dJ_c\|_2 \leq tolG(1 + \mid J_c \mid)$

   $Stop(4) = \|dJ_c\|_2 \leq 10^{-6}\ eps$

   $Stop(5) = iter \geq maxIter$

   Si les trois premières conditions sont vérifiées, ou la quatrième condition est vérifiée ou la cinquième condition est vérifiée, alors $arret = 1$
3. Appeler une procédure de résolution de type GC pour le système :

   $$H_c\ dy = -dJ_c$$
4. Appeler la procédure de recherche linéaire Armijo avec dy_c pour obtenir y_t respectant la longueur de pas
5. Stockage et mise à jour des valeurs :

   $y_{Old} = y_c$, $J_{Old} = J_c$, $y_c = y_t$

   Mise à jour et évaluation de J_c, dJ_c et H_c en y_c

end (while)

4.2 Stratégie de résolution

Dans le processus du recalage d'images, nous avons vu que la phase d'optimisation utilise l'algorithme de Gauss-Newton pour résoudre un problème aux moindres carrés non linéaires. Cette méthode utilise un processus de linéarisation tel que nous l'avons vu à la section 2.3. A chaque itération externe i correspond un système linéaire symétrique défini positif de la forme $A^{(i)}x = b^{(i)}$. Pour le résoudre, nous pouvons appliquer l'algorithme du gradient conjugué. Afin d'accélérer la résolution de ces systèmes, la suite de ce chapitre va développer une stratégie intégrant les préconditionneurs LMP et utilisant le gradient conjugué préconditionné.

Si, d'une itération externe à l'autre, la matrice des systèmes ne varie pas significativement, les vecteurs A-conjugués collectés lors de la résolution par la méthode du gradient conjugué peuvent être utilisés pour préconditionner les systèmes suivants. En effet, dans la section 3.4, nous avons montré que les trois instances LMP proposées restent pertinentes pour préconditionner des systèmes perturbés dans une certaine mesure.

La stratégie développée est la suivante. Premièrement, nous allons résoudre le système $A^{(1)}x = b^{(1)}$ à l'aide de l'algorithme du gradient conjugué durant lequel nous collecterons des vecteurs A-conjugués produits de son exécution. Ces vecteurs sont les directions conjuguées ou les paires de Ritz obtenues par la proposition 2.10. Ces informations sont utilisées afin d'accélérer la convergence des systèmes suivants. Ceux-ci seront préconditionnés par l'une des trois instances de la classe LMP que nous avons proposées précédemment et dont les expressions des formes factorisées sont, dans l'ordre, le spectral-LMP, le Ritz-LMP et le quasi-Newton-LMP

$$G_l = \prod_{i=1}^{l} \left[I_n - \left(1 - \frac{1}{\sqrt{\lambda_i}}\right) v_i v_i^T \right],$$

$$G_l = \prod_{i=1}^{l} \left[I_n - \left(1 - \frac{1}{\sqrt{\theta_i}}\right) z_i z_i^T - \frac{e_l^T y_i \eta_l}{\theta_i} z_i q_{l+1}^T \right],$$

$$G_l = \prod_{i=1}^{l} \left[I_n - p_l \left(\frac{p_l^T A}{p_l^T A p_l} + \frac{1}{\sqrt{p_l^T A p_l}} \frac{r_l^T}{\|r_l\|_2} \right) \right],$$

avec (λ_i, v_i) les paires propres, (θ_i, z_i) les paires de Ritz, q_{k+1} vecteur du processus de Lanczos, p_i les directions de descente, r_i les résidus de la résolution par le gradient conjugué, l le nombre d'itérations dans le gradient conjugué et $i = 1, \ldots, l$. Pour rappel, en pratique, les paires propres (λ_i, v_i) seront approximées par les paires de Ritz (θ_i, z_i). Une fois le préconditionneur LMP désiré construit, les systèmes $A^{(i)}x = b^{(i)}$, avec $i > 1$, vont être résolus à l'aide du gradient conjugué préconditionné utilisant une des trois instances LMP.

L'étape 3 de l'algorithme 4.1 devient :

Appeler une procédure de résolution de type GC pour le système :

$$H_c \, dy = -dJ_c$$

if $iter == 1$

1. Résoudre via GC et collecter de l'information
2. Construire G un préconditionneur LMP avec l'information récoltée

else

Résoudre via GCP utilisant G

end

4.3 Expérimentations numériques

Dans cette section, nous allons considérer le recalage des deux images de la figure 4.3. Lors du processus de recalage dans FAIR, la stratégie de résolution présentée dans la section précédente va être mise en place. Les résultats obtenus vont être analysés et les préconditionneurs LMP classés selon leur performance.

Variation du Hessien

La stratégie présentée se base sur la variation non significative du Hessien. La validité de cette hypothèse peut être vérifiée en examinant les valeurs de Ritz entre les itérations externes [27]. La figure 4.6 (a) illustre les valeurs de Ritz triées de manière croissante et collectées grâce à 70 itérations du gradient conjugué et ce pour les 3 premiers systèmes linéaires. Nous observons que les valeurs de Ritz du premier système ne sont pas toujours proches de celles des deux autres systèmes.

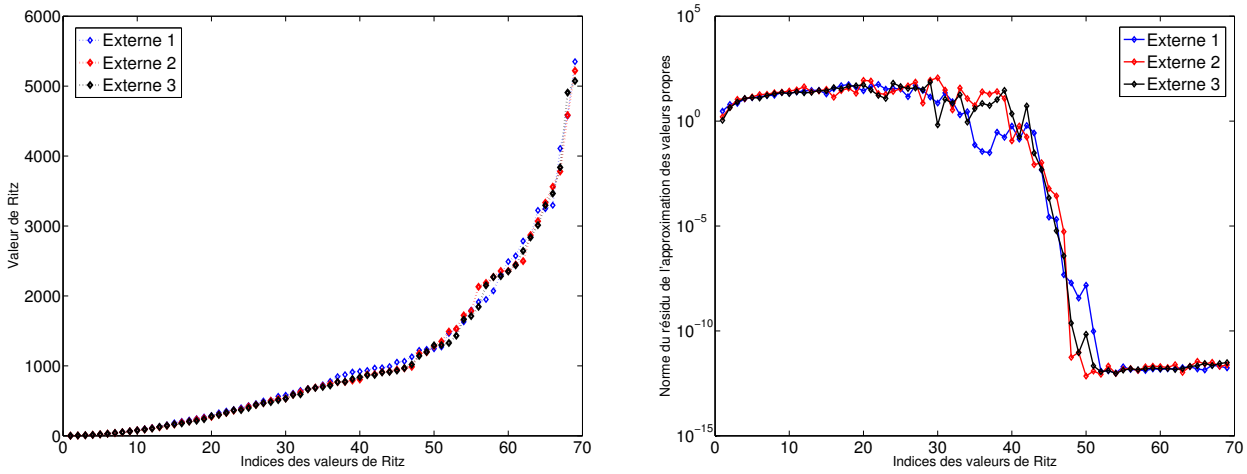
Le spectral-LMP est construit à partir des valeurs de Ritz, il est donc pertinent de regarder à la qualité de l'approximation. Pour chaque paire de Ritz (θ_i, z_i), nous calculons la norme du résidu de l'approximation

$$\| \hat{r}_i \|_2 = \| A z_i - \theta_i z_i \|_2 \tag{4.4}$$

et nous obtenons la figure 4.6 (b). Pour les trois systèmes, l'extrémité supérieure des valeurs de Ritz a convergé vers les valeurs propres.

Nous avons appliqué la stratégie développé précédemment et le nombre de conditionnement de la matrice du second et du troisième système est détérioré après préconditionnement. Dans cet exemple de recalage, l'hypothèse faite sur le Hessien (4.3) ne semble pas être vérifiée. En effet, les valeurs de Ritz de la première itération externe ne sont pas tout à fait semblables à celles des deux autres itérations, qui elles sont proches entre elles. Sur les deux figures, les courbes noire et rouge sont proches tandis que la bleue ne l'est pas en certains points.

Pour pallier à ce problème, le Hessien peut être approximé uniquement par (4.2). Le terme (4.1) relatif au critère de dissimilarité varie d'un système à l'autre ce qui n'est pas le cas de (4.2). De par cette approximation, chaque système possède le même Hessien puisque celui-ci dépend de l'opérateur différentiel et varie l'un par rapport à l'autre uniquement par leur second membre.



(a) (b)

FIGURE 4.6 – Valeurs de Ritz et résidus de l'approximation des valeurs propres pour les trois premiers systèmes non préconditionnés.

Après cette modification dans l'approximation du Hessien, celui-ci est représenté à la figure 4.7. Dans la résolution associée au niveau 4, la matrice est de dimension 544 et son conditionnement vaut 2.3549 10^4. La matrice est composée de 4636 / 544^2 = 1.56% d'éléments non nuls.

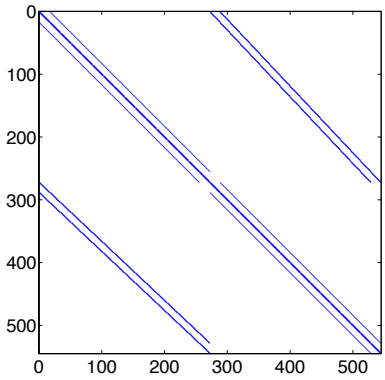
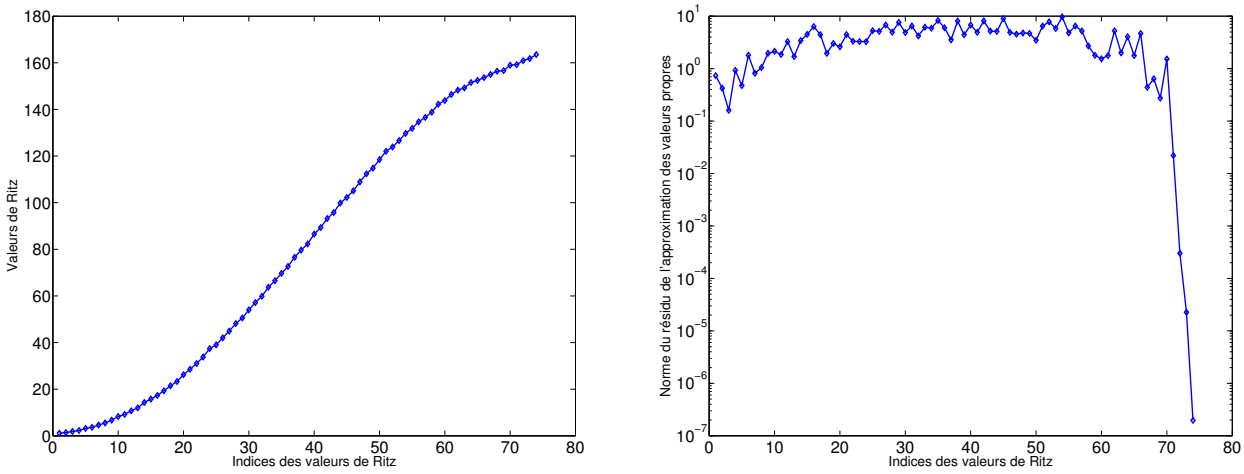


FIGURE 4.7 – Représentation des éléments non nuls du Hessien.

La résolution du système associé à la première itération externe va permettre d'obtenir l'information adéquate pour construire les trois instances LMP. La figure 4.8 (a) illustre les 74 valeurs de Ritz triées de manière croissante et collectées grâce aux itérations du gradient conjugué. La figure 4.8 (b) permet d'observer qu'une petite partie de l'extrémité supérieure des valeurs de Ritz approxime le mieux les valeurs propres de la matrice du système.



(a) (b)

FIGURE 4.8 – Valeurs de Ritz et résidus de l'approximation des valeurs propres pour le premier système.

Classement des instances

La résolution du recalage des deux images par la méthode de Gauss-Newton comporte 7 itérations externes. L'objectif est de tester et classer les 3 préconditionneurs LMP. Leurs performances sont uniquement comparables sur la seconde itération externe. En effet, la première résolution est commune puisqu'il s'agit du gradient conjugué permettant de collecter l'information pour la construction des préconditionneurs. En pratique, le point de départ du premier système est la solution du pré-recalage rigide. Quant au second système, il est identique car le même problème quadratique est résolu. A partir du troisième système, les solutions obtenues sont différentes et le processus de linéarisation détermine des systèmes quadratiques différents dépendant de la solution du second système.

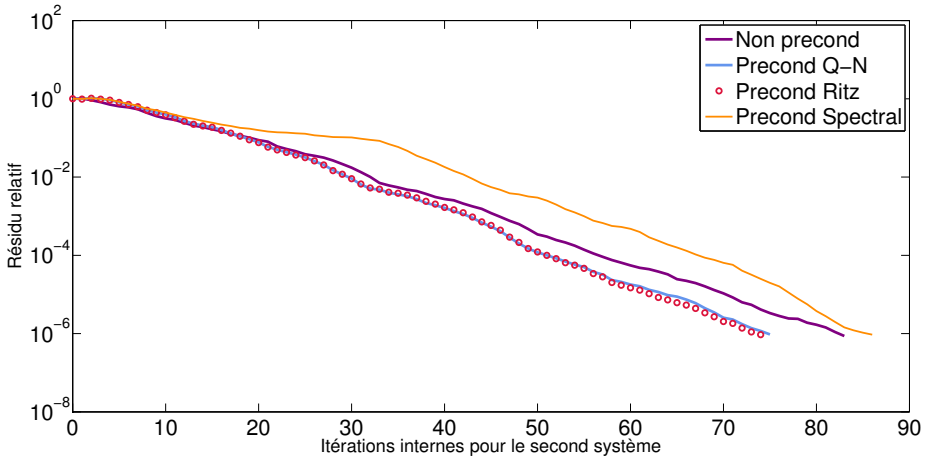


FIGURE 4.9 – Comparaison des 3 préconditionneurs LMP sur la seconde itération externe.

Analysons l'amélioration en terme d'itérations à l'aide de la figure 4.9 et la table 4.2. Pour le second système, la résolution avec le quasi-Newton-LMP a pris une itération de plus qu'avec le Ritz-LMP. Les deux instances permettent respectivement un gain de 8 et 9 itérations.

Nous observons que la convergence se détériore lors du préconditionnement par le spectral-LMP. Cela s'explique par le fait que toutes les valeurs de Ritz ne sont pas des bonnes approximations de l'information spectrale de la matrice à préconditionner. L'information spectrale étant inexacte, le spectral-LMP tend à perdre ses propriétés de préconditionneurs et n'est donc plus véritablement un préconditionneur LMP.

Préconditionneur	Par itération externe							Total
	1	2	3	4	5	6	7	
Aucun	74	83	82	83	82	81	80	565
Quasi-Newton-LMP	74	75	73	71	75	73	75	516
Ritz-LMP	74	74	73	74	76	74	75	520
Spectral-LMP	74	86	85	85	85	86	84	585

TABLE 4.2 – Table de comparaison des 3 préconditionneurs LMP au fil des itérations externes.

Globalement, pour chaque système résolu, le spectral-LMP nécessite plus d'itérations que la résolution sans préconditionnement. Le quasi-Newton-LMP totalise le plus petit nombre d'itérations internes cumulées bien que la différence avec le Ritz-LMP ne soit que de 4 itérations.

Le spectral-LMP avec de l'information spectrale exacte

Nous venons de voir que le spectral-LMP donne de moins bons résultats que les deux autres instances proposées. Il serait donc intéressant de sélectionner des paires de Ritz ayant suffisamment convergé vers les paires propres de la matrice. En pratique, cela revient à sélectionner les paires pour lesquelles (4.4) est inférieur à une tolérance fixée.

De ce fait, le Ritz-LMP peut être considéré comme une version stabilisée et enrichie du spectral-LMP. En effet, il utilise toutes les paires de Ritz disponibles sans se soucier du problème de convergence que le spectral-LMP rencontre.

Une autre possibilité est la construction du spectral-LMP avec de l'information spectrale exacte. Il est aisé d'obtenir un nombre déterminé de paires propres exactes à l'aide de la fonction `eigs` de Matlab. Nous en récupérons 74 afin de construire le préconditonneur spectral-LMP avec la formule (3.9) en utilisant la même quantité d'information que précédemment. Le second système est résolu avec le gradient conjugué préconditionné utilisant ce préconditionneur et les résultats sont illustrés à la figure 4.10. L'information spectrale exacte permet d'obtenir un meilleur résultat avec le passage de 83 à 67 itérations.

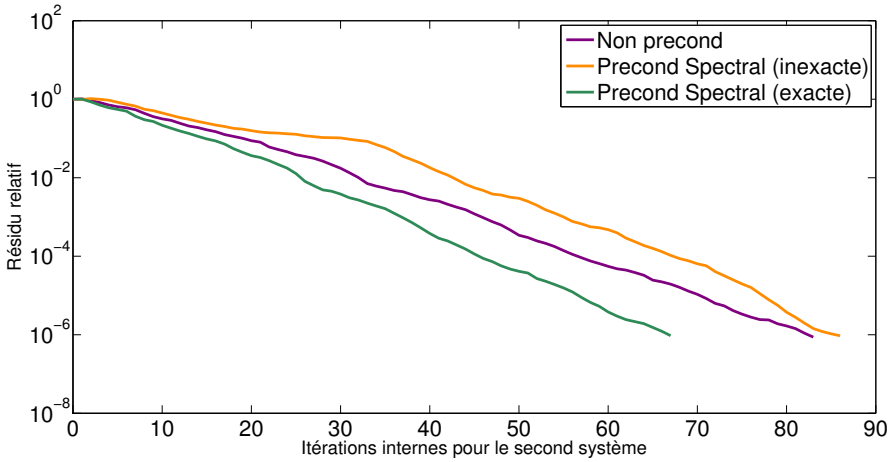


FIGURE 4.10 – Comparaison du préconditionneur spectral-LMP construit avec de l'information spectrale exacte et inexacte sur la seconde itération externe.

4.4 Limitation

Cette section va mettre en avant une limitation des LMP dans le cadre du recalage d'images dans FAIR. Le chapitre précédent précisait que plus la taille de la matrice hessienne augmente, plus le nombre de vecteurs impliqués dans la construction du préconditionneur LMP doit être important. Or, le nombre de vecteurs collectés dépend du nombre d'itérations internes pour que la solution du premier système converge.

Utilisons la discrétisation de niveau 5 des images proposée à la figure 4.4. La matrice hessienne est maintenant de dimension 2112. En appliquant la même stratégie qu'à la section précédente, nous obtenons la figure 4.11. Tout comme précédemment, nous observons que le spectral-LMP détériore aussi la convergence pour ce système. Le Ritz-LMP et le quasi-Newton-LMP n'améliorent pas significativement la convergence. En effet, la résolution du premier système n'a permis de récolté que 85 vecteurs, ce qui est très peu comparé à la taille du système. Il est donc cohérent d'avoir une si faible amélioration.

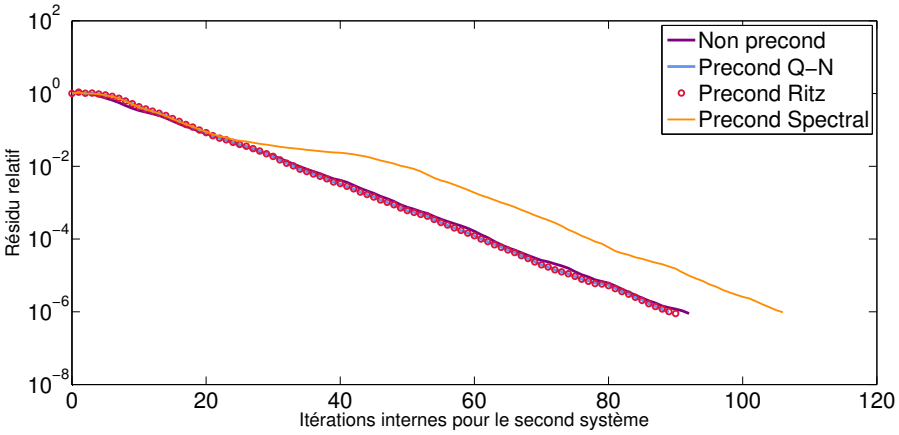


FIGURE 4.11 – Comparaison des 3 préconditionneurs LMP sur la seconde itération externe.

Il est possible d'augmenter le nombre de vecteurs en diminuant la tolérance du critère d'arrêt sur le résidu relatif. Cependant, réduire excessivement cette tolérance pour les systèmes de grande taille n'est pas acceptable. Si le niveau de discrétisation des images est de plus en plus fin, la taille du système à résoudre va augmenter. Une trop petite tolérance va engendrer un grand nombre d'itérations. Le temps de calcul devient alors trop important et l'information à stocker est considérable. Initialement, FAIR fixait cette tolérance à 10^{-1} et nous l'avons changée pour imposer 10^{-6} afin de collecter un plus grand nombre de vecteurs. En diminuant ce critère, nous avons amélioré l'estimation de la solution. Or, la résolution du problème de recalage ne nécessite pas une solution aussi fine. Nous gaspillons du temps de calcul pour obtenir de l'information qui permettra de préconditionner et dans l'application proposée, le préconditionnement améliore très peu la convergence. Ainsi, lorsque le niveau de discrétisation augmente, il parait judicieux de se rabattre sur d'autres méthodes de préconditionnement telles que les méthodes de Jacobi ou de Gauss-Seidel.

4.5 Conclusion

Dans ce chapitre, nous avons étudié la résolution numérique d'un problème de recalage d'images dans FAIR. Dans un premier temps, nous avons vu qu'en pratique FAIR procède à un recalage multi-niveaux. Pour un niveau fixé, nous avons illustré l'importance du pré-recalage. Ensuite, le fonctionnement de l'algorithme de Gauss-Newton dans FAIR a été abordé.

Nous avons élaboré une stratégie de résolution afin d'introduire les LMP dans l'algorithme existant. La méthodologie mise en place consiste à résoudre le premier système à l'aide de l'algorithme du gradient conjugué. Lors de cette résolution, les directions de descente et les paires de Ritz sont collectées. Ainsi, les trois instances LMP peuvent être construites et utilisées dans la résolution des systèmes suivants à l'aide de l'algorithme du gradient conjugué préconditionné. Cette stratégie impose que le Hessien ne varie pas significativement d'une itération externe à l'autre. Dans l'application choisie, nous avons dû faire face à une trop grande variation de celui-ci. Pour pallier au problème, nous avons choisi de l'approximer par le Hessien du terme relatif à la régularisation. Ainsi, la matrice hessienne de chaque système est identique et seul le membre de droite varie entre les itérations externes.

Les performances des trois instances ont pu être classées et ont mis en avant le Ritz-LMP et le quasi-Newton-LMP. Quant au spectral-LMP, nous avons observé que les paires de Ritz n'ont pas suffisamment convergé vers les paires propres. Son résultat n'est pas satisfaisant dans le cadre de notre application mais sa construction avec de l'information exacte donne des résultats prometteurs. De plus, cette application concrète a permis de mettre en évidence la limitation des LMP dans le cas d'une discrétisation plus fine des images. Nous pouvons conclure que le gain en terme d'itérations dépend de la quantité d'informations collectées mais aussi de sa qualité.

Conclusions et perspectives

Ce mémoire a commencé par définir la problématique de l'imagerie médicale et, plus particulièrement, le recalage d'images médicales. Il s'agit d'un processus de traitement d'images qui consiste à placer deux images dans un référentiel commun afin de pouvoir combiner et/ou comparer leurs informations. Nous avons vu que le problème du recalage d'images pouvait s'écrire comme un problème d'optimisation qui permet de trouver la transformation spatiale qui minimise la distance entre deux images. Cette distance est définie par un critère de dissimilarité que nous avons choisi comme la somme des carrés des différences des intensités.

Dans le cas du recalage non rigide, la transformation permet de déformer l'image, contrairement au recalage rigide, qui consiste, par exemple, à effectuer des rotations et des translations. Le terme de régularisation permet de choisir le type de déformation à appliquer à l'image. Ainsi, la fonction objectif du problème d'optimisation est composée du critère de dissimilarité et du terme relatif à la régularisation. Ce problème d'optimisation s'apparente à un problème des moindres carrés non linéaires qui peut être résolu à l'aide de l'algorithme de Gauss-Newton. Cette méthode est une résolution itérative qui, à chaque itération, utilise un processus de linéarisation et résout un système linéaire symétrique défini positif. Un outil très efficace dans la résolution de ce type de système est l'algorithme du gradient conjugué que nous avons présenté dans le second chapitre. La convergence de l'algorithme peut être améliorée grâce au préconditionnement.

Le troisième chapitre s'intéresse à une classe de préconditionneurs dits préconditionneurs à mémoire limitée (LMP) dont nous avons présenté les propriétés principales. Nous nous sommes intéressés à trois instances de cette classe, à savoir, le spectral-LMP, le Ritz-LMP et le quasi-Newton-LMP. Ces préconditionneurs ont la particularité d'être construits à partir de vecteurs linéairement indépendants et conjugués par rapport à la matrice à préconditionner. L'attrait pour ces préconditionneurs réside dans le fait que les vecteurs qui les composent sont automatiquement produits dans l'algorithme du gradient conjugué. Ils peuvent donc être construits et appliqués à un coût relativement faible.

En pratique, les vecteurs utilisés dans le quasi-Newton-LMP sont les directions conjuguées, alors que le spectral-LMP et le Ritz-LMP se basent sur les paires de Ritz. Nous avons vu que le Ritz-LMP est une formulation plus générale du spectral-LMP. Avant de les appliquer au problème d'optimisation lié au recalage d'images, nous avons expérimenté les trois instances sur des systèmes à second membres multiples. L'idée est d'utiliser les vecteurs collectés lors des résolutions précédentes par le gradient conjugué afin de construire les préconditionneurs et de les utiliser dans la résolution des systèmes suivants. Les résultats ont confirmé que si toute l'information disponible est utilisée alors le Ritz-LMP et le quasi-Newton-LMP sont analytiquement équivalents. Sur les exemples proposés, nous avons observé que la performance augmente avec le nombre de vecteurs impliqués dans la construction des préconditionneurs.

Nous nous sommes placés dans le contexte de résolution de systèmes perturbés en faisant l'hypothèse que les matrices hessiennes ne changent pas significativement. Nous avons montré que l'utilisation du quasi-Newton-LMP donne de bons résultats même si ceux-ci sont moins satisfaisants que précédemment. Cependant, dans la littérature, il n'est pas spécifié à quel point le LMP reste efficace si la matrice hessienne varie. Nous ne disposons que d'analyses empiriques pour lesquelles les matrices ne varient pas assez que pour atténuer l'effet positif du LMP.

Pour terminer, le dernier chapitre a étudié la résolution numérique d'un problème de recalage d'images médicaledans la librairie FAIR [1]. Nous avons présenté le fonctionnement de l'algorithme de Gauss-Newton dans FAIR auquel nous avons adapté une stratégie afin d'intégrer les LMP à la résolution. Cette stratégie consiste à résoudre le premier système à l'aide de l'algorithme du gradient conjugué et ainsi collecté les directions de descente et l'information de Ritz. Un des préconditionneurs LMP est alors construit et utilisé dans les résolutions suivantes. Cette stratégie s'appuie sur la variation non significative du Hessien. Cependant, dans l'application que nous avons choisie, le Hessien varie trop d'un système linéaire à l'autre que pour bénéficier correctement de l'effet du LMP. Afin de pallier à ce problème, nous avons approximé le Hessien par le Hessien du terme relatif à la régularisation de la fonction objectif. Ainsi, la matrice de chaque système à résoudre est identique et seul le membre de droite varie.

A un faible niveau de discrétisation des images, le Ritz-LMP et le quasi-Newton-LMP ont permis d'améliorer la convergence. Quant au spectral-LMP construit à partir de l'information de Ritz, il a détérioré le nombre de conditionnement de la matrice du système. En cause, les paires de Ritz qui n'ont pas suffisamment convergé vers les paires propres. La construction du spectral-LMP avec de l'information spectrale exacte donne de bons résultats. Il serait intéressant d'utiliser des paires propres obtenues par les formules trigonométriques relatives au calcul des vecteurs propres et valeurs propres des matrices de Toeplitz (voir [33, chapitre 7]).

Nous avons observé que le gain en terme d'itérations dépend de la qualité de l'information collectée mais aussi de sa quantité. Le nombre de vecteurs dépend du nombre d'itérations lors de la résolution du première système. Si le niveau de discrétisation des images est plus fin, la taille du système à résoudre augmente et le nombre de vecteurs doit donc être plus important. En réduisant la tolérance du critère d'arrêt sur le résidu relatif, nous collectons plus de vecteurs. Or, utiliser une tolérance trop petite va nécessiter un grand nombre d'itérations. Le temps de calcul va s'allonger et la quantité d'information à stocker va augmenter considérablement. L'algorithme dans FAIR, utilise initialement une tolérance de 10^{-1} et un nombre maximum de 500 itérations. Aller au-delà de ce critère n'est pas utile dans la résolution du problème mais bien dans la création du préconditionneur.

Bibliographie

[1] Flexible Algorithms for Image Registration : https://github.com/C4IR/FAIR.m.

[2] MALANDAIN G. : *Les mesures de similarité pour le recalage des images médicales*. Habilitation à diriger des recherches, Université de Nice Sophie-Antipolis, France, Mars 2006.

[3] NOBLET V. : *Recalage non rigide d'images cérébrales 3D avec contrainte de conservation de la topologie*. Thèse de doctorat, Université Louis Pasteur, Strasbourg, France, Mars 2006.

[4] Wikipédia, imagerie médicale : https://fr.wikipedia.org/wiki/Imagerie_m%C3%A9dicale, consulté le 13 Mai 2017.

[5] Librairie Médicale, *http ://www.remede.org/librairie-medicale/pdf/e9782729861049.pdf*, consulté le 13 Mai 2017.

[6] Psychiatrie Neurologie Gériatrie Poitiers, http://www.psychiatrie-neurologie-geriatrie-poitiers.fr/wp-content/uploads/2014/11/Image21.jpg, Novembre 2014, consulté le 13 Mai 2017.

[7] FORREST W. : *Simultaneous PET/MRI creates diagnostic-quality images*, http://www.auntminnie.com. 2010, consulté le 25 Avril 2016.

[8] Wikipédia : *Imagerie médicale* , http://math.nist.gov/MatrixMarket/matrices.html, consulté le 26 Avril 2016.

[9] ROCHE A. : *Recalage d'images médicales par inférence statistique*. Thèse de doctorat, Université de Nice Sophia Antipolis, France, 2001.

[10] HAJNAL J.V., HILL D.L. G. et HAWKES D. J. : *Medical image registration*. RAND Corporation, London, CRC Press, 2001.

[11] PETITJEAN C. : *Le recalage d'images*, http://carolinepetitjean.free.fr/enseignements/ti/coursrecalage$_$petitjean.pdf. 2012, consulté le 27 Avril 2016.

[12] ATIF J. : *Recalage non-rigide multimodal des images radiologiques par information mutuelle quadratique normalisée*. Thèse de doctorat, Université de Paris XI, France, 2004.

[13] DELAHAUT C. : *Médecine nucléaire : optimisation dans le cadre du recalage d'images médicales*. Mémoire de fin d'études, FUNDP Namur, Belgique, Juin 2014.

[14] CACHIER P. : *Recalage non rigide d'images médicales volumiques : contributions aux approches iconiques et géométriques*. Thèse de doctorat, Ecole Central Paris, France, 2002.

[15] MODERSITZKI J. : *FAIR - Flexible algorithms for image registration*. Vol. 6, SIAM, 2009.

[16] MODERSITZKI J. : *Numerical Methods for Image Registration*. Oxford University Press, 2004.

[17] OZE S. : *Modélisation mathématique de problème relatifs au recalage d'images*. Thèse de doctorat, INSA Rouen, France, Novembre 2015.

[18] NOCEDAL J. et WRIGHT S. J. : *Numerical Optimization*,seconde édition. Springer New York, 2006.

[19] SAAD Y. : *Iterative Methods for Sparse Linear Systems*, seconde édition. SIAM, 2003.

[20] KAUTH D. : *Optimisation numérique : Méthodes du gradient conjugué*, chapitre 5.1. Université de Fribourg, 2009.

[21] TSHIMANGA J. : *On a class of limited memory preconditioners for large-scale nonlinear least-squares problems (with application to variational ocean data assimilation)*. Thèse de doctorat, FUNDP Namur, Belgique, 2007.

[22] BUHENDWA NYENYEZI J. : *Développement et étude d'algorithmes d'optimisation pour la résolution numérique de problèmes issus de l'imagerie médicale*. Epreuve de confirmation, Université de Namur, Belgique, Août 2015.

[23] QUARTERONI A., SACCO R. et SALERI F. : *Méthodes numériques : Algorithmes, analyse et applications*, seconde édition. Springer Science and Business Media, 2008.

[24] GOLUB. G.H. et VAN LOAN C.F. : *Matrix Computation*, quatrième édition. Johns Hopkin University Press, Baltimore, 2013.

[25] GOLUB. G.H. et VAN LOAN C.F. : *Matrix Computation*, troisième édition. Johns Hopkin University Press, Baltimore, 1996.

[26] TANNIER C. : *Etudes des algorithmes de Lanczos et du gradient conjugué et leurs applications à la résolution de systèmes à seconds membres multiples*. Mémoire de fin d'études, FUNDP Namur, Belgique, Juin 2009.

[27] TSHIMANGA J., GRATTON S., WEAVER A. T. et SARTENAER A. : *Limited-memory preconditioners, with application to incremental four-dimensional variational data assimilation*. Quarterly Journal of the Royal Meteorological Society 134 : 751–769, 2008.

[28] GRATTON S., SARTENAER A. et TSHIMANGA J. : *On a class of limited memory preconditioners for large scale linear systems with multiple right-hand sides*. SIAM Journal on optimization, 21(3), 912–935, 2011.

[29] BYRD R.H., NOCEDAL J. et SCHNABEL R.B. : *Representations of quasi-Newton matrices and their use in limited memory methods*. Mathematical Programming, 63(1) : 129–156, 1994.

[30] Harwell Boeing Sparse Matrix Collection : `http://math.nist.gov/MatrixMarket/matrices.html`.

[31] MORALES J. L. et NOCEDAL J. : *Automatic preconditioning by limited memory quasi-newton updating*. SIAM Journal on Optimization 10(4) : 1079–1096, 2000.

[32] BATY X. : *Recalage de séquences cardiaques spation-temporelles IRM et TEP/SCAN*. Thèse de doctorat, Université d'Angers, France, Juillet 2007.

[33] MEYER C. D. : *Matrix analysis and applied linear algebra*. Vol. 2, SIAM, 2000.

Annexes

Nous présentons dans ces annexes les codes Matlab qui ont servis à réaliser les analyses des chapitres 3 et 4. Ils sont insérés dans l'ordre suivant.

- Les codes pour construire les 3 préconditionneurs LMP.
- Le code de l'algorithme du gradient conjugué permettant de collecter l'information pour construire les trois instance LMP.
- Le code `GaussNewton.m` de la librairie FAIR [1] modifié pour intégrer la stratégie présentée dans le dernier chapitre.

```
%-------------------------------------------------------------------------
% Décembre 2016
%
% Objectif : construire la forme factorisée du spectral-LMP
%
% INPUT : (lambda,V) : paires propres de la matrice à préconditionner
%
% OUTPUT : G : matrice de préconditionnement, spectral-LMP
%-------------------------------------------------------------------------

function G = ConstrucSpecFacto(lambda,V)

[n,l] = size(V);
G = eye(n);

for i = 1 : l
    G = (eye(n) - (1 - 1/sqrt(lambda(i))) * V(:,i) * V(:,i)') * G;
end
end
```

```
%-------------------------------------------------------------------------
% Janvier 2017
%
% Objectif : construire la forme factorisée du Ritz-LMP
%
% INPUT :
% (theta,Y) : paires propres de la matrice T de l'algorithme de Lanczos
% eta : scalaires de l'algorithme de Lanczos
```

```
% q : vecteur de l'algorithme de Lanczos
% Z : vecteurs de ritz
%
% OUTPUT : G : matrice de préconditionnement, Ritz-LMP
%-------------------------------------------------------------------------

function G = ConstrucRitzFacto(theta,eta,q,Z,Y)

n = length(q);
l = length(Y);
G = eye(n);
e = eye(l);

for i = 1 : l
    w = (e(:,l)'* Y(:,i) * eta) / theta(i);
    G = (eye(n) - (1 - 1/sqrt(theta(i))) * Z(:,i) * Z(:,i)' - w *
        Z(:,i) * q') * G;
end
end
```

```
%-------------------------------------------------------------------------
% Janvier 2017
%
% Objectif : construire la forme factorisée du quasi-Newton-LMP
%
% INPUT :
% s : directions conjuguées
% As : vecteurs multipliés par la matrice à préconditionner
% res : résidus obtenus lors de la résolution du problème A x = b
%       par l'algorithme du gradient conjugué
%
% OUTPUT : G : matrice de préconditionnement, quasi-Newton-LMP
%-------------------------------------------------------------------------

function G = ConstrucQNFacto(s,As,res)

[n,l] = size(s);
G = eye(n);

for i = 1 : l
    tmp = s(:,i)'*As(:,i);
    G = (eye(n) - s(:,i)* (As(:,i)'/tmp + res(:,i)' / (sqrt(tmp) *
        norm(res(:,i))))* G;
end
end
```

```
%-------------------------------------------------------------------------
% Décembre 2016
% Algorithme du Gradient Conjugué
%
% Objectif : Méthode itérative résolvant le système d'équations A x = b.
%            Construction des paires de Ritz, collecte d'information
%            afin de construire les 3 instances LMP : quasi-Newton-LMP,
%            Ritz-LMP et spectral-LMP
%
% INPUT :
% A : matrice symétrique définie positive
% b : membre de droite
% maxit : nombre maximum d'itérations
% tol : tolérance pour le critère d'arrêt sur le résidu relatif
%
% OUTPUT :
% result : solution du problème A x = b
% iteration : nombre d'itérations dans la boucle
% r : résidu
% p : directions conjuguees
% Ap : directions conjuguées multipliées par la matrice
%      à préconditionner
% normr : norme 2 du résidu
% (theta,Y) : paires propres de la matrice T, tridiagonale de Lanczos
% Z : vecteurs de Ritz
% etaout : eta_l scalaire de l'algorithme de Lanczos
% Qout : vecteur q_{k+1} de l'algorithme de Lanczos
%-------------------------------------------------------------------------

function [result,iteration,r,p,Ap,normr,Qout,Y,Z,theta,etaout] =
    GradConj(A,b,maxit,tol)

[n,m] = size(A); % Taille du système
x = zeros(n,1);  % Point de départ : vecteur nul

r(:,1) = b - A*x;          % Résidu
normr(1) = norm(r(:,1));   % Norme 2 du résidu
p(:,1) = r(:,1);           % Initialisation de la 1ere direction
    conjuguée
Ap(:,1) = A*p(:,1);
normb = norm(b);

j = 1;
while (normr(j) >= tol*normb) && (maxit >= j)

    alpha(j) = (r(:,j)'*r(:,j)) / (p(:,j)'*Ap(:,j));
```

```
45      x = x + alpha(j) * p(:,j);
46      r(:,j+1) = r(:,j) - alpha(j) * Ap(:,j);
47
48      %---- Orthogonalisation des résidus :
49      v = r;
50      [m,nn] = size(v); Qo = zeros(m,nn); R = zeros(n,nn);
51
52      for i = 1 : nn
53          R(i,i) = norm(v(:,i));
54          Qo(:,i) = v(:,i)/R(i,i);
55          for h = i+1 : nn
56              R(i,h) = Qo(:,i)'*v(:,h);
57              v(:,h) = v(:,h) - Qo(:,i) * R(i,h);
58          end
59      end
60      r(:,j+1) = v(:,j+1);
61
62      clear v; clear Qo; clear R;
63      %----
64
65      beta(j) = (r(:,j+1)'*r(:,j+1)) / (r(:,j)'*r(:,j));
66      p(:,j+1) = r(:,j+1) + beta(j) * p(:,j);
67      Ap(:,j+1) = A*p(:,j+1);
68      normr(j+1) = norm(r(:,j+1));
69      j = j+1;
70  end
71
72  result = x;
73  j = j-1;
74  iteration = j;
75
76  % -----  Relation Gradient Conjugué et Algorithme de Lanczos
77  % Construction de T :
78  diago = zeros(j,1);
79  horsdiad = zeros(j-1,1);
80
81  diago(1) = 1 / alpha(1);
82  for i = 2 : jw
83      diago(i) = 1 / alpha(i) + beta(i-1) / alpha(i-1);
84  end
85
86  for i = 1 : j-1
87      horsdiag(i) = sqrt(beta(i)) / alpha(i);
88  end
89
90  T = gallery('tridiag',horsdiag,diago,horsdiag);
```

```
91  T = T + zeros(j,j);
92
93  % Construction de Q :
94  Q = zeros(n,j);
95  delta = zeros(j,1);
96  for i = 1 : j
97      delta(i) = norm(r(:,i));
98      Q(:,i) = (-1)^(i+1) * r(:,i) / delta(i);
99  end
100
101 % Valeurs propres et vecteurs propres de T :
102 [Y,D] = eig(T);
103 theta = diag(D);
104
105 % Vecteurs de Ritz :
106 Z = Q*Y;
107
108 % Vecteur et scalaire de l'algorithme de Lanczos impliqués dans la
        construction du préconditionneur Ritz-LMP :
109 Qout = (-1)^(j+1+1) * r(:,j+1)/norm(r(:,j+1));
110 etaout = sqrt(beta(j)) / alpha(j);
111 end
```

```
1
2  %-------------------------------------------------------------------------
3  % Février 2017
4  % Algorithme de Gauss-Newton
5  % Code de la libraire FAIR modifié pour intégrer
6  % les trois préconditionneurs LMP :
7  % Quasi-Newton-LMP, Ritz-LMP et spectral-LMP
8  %
9  %=========================================================================
10 % This code is part of the Matlab-based toolbox
11 % FAIR - Flexible Algorithms for Image Registration.
12 % For details see
13 % - https://github.com/C4IR and
14 % - http://www.siam.org/books/fa06/
15 %=========================================================================
16 % (c) Jan Modersitzki 2010/12/27, see FAIR.2 and FAIRcopyright.m.
17 % http://www.mic.uni-luebeck.de/people/jan-modersitzki.html
18 %
19 % function [yc,His] = GaussNewtonLMP(fctn,yc,varargin)
20 %
21 % Gauss-Newton scheme with variable line search
22 % for minimizing J = fctn(yc)
23 %
24 % Input :
25 %   fctn        function handle
26 %   yc          starting guess
27 %   varargin    optional parameter, see below
28 %
29 % Output :
30 %   yc          numerical optimizer (current iterate)
31 %   his         iteration history
32 %=========================================================================
33
34 function [yc,His] = GaussNewtonLMP(fctn,yc,varargin)
35
36 % -- parameter initialization
37 lineSearch   = @Armijo;         % default line search
38 maxIter      = 10;              % max number of iterations
39 tolJ         = 1e-3;            % for stopping, objective function
40 tolY         = 1e-2;            %   - " -     , current value
41 tolG         = 1e-2;            %   - " -     , norm of gradient
42 LSMaxIter    = 10;              % maximum number of line search
      iterations
43 LSreduction  = 1e-4;            % minimal reduction in line search
44 vecNorm      = @norm;           % norm to be used for dJ and dy
45 yStop        = [];     % used for stopping in multi-level framework
```

```
Jstop         = [];
Plots         = @(iter,para) []; % for plots;

% ------- Choix du préconditionneur LMP :
LMP = 'QN';
%LMP = 'Ritz';
%LMP = 'Spectral';
% -------

for k=1:2:length(varargin),     % overwrites default parameter
    eval([varargin{k},'=varargin{',int2str(k+1),'};']);
end;

if ~isa(Plots,'function_handle') && (Plots == 0 || strcmp(Plots,'off')),
    Plots = @(iter,para) []; % for plots;
end;

if isempty(yStop), yStop  = yc; end; % yStop used for stopping only
% -- end parameter setup

% some output
FAIRmessage = @(str) fprintf('%% %s  [ %s ]  % s\n',
   char(ones(1,10)*'-'),str,char(ones(1,60-length(str))*'-'));
fprintf('[ maxIter=%s / tolJ=%s / tolY=%s / tolG=%s / length(yc)=%d
   ]\n',
        num2str(maxIter),num2str(tolJ),num2str(tolY),num2str(tolG),length(yc));

% -- initialize
STOP = zeros(5,1); % Critères d'arrêt

if isempty(Jstop),
% evaluate objective function for stopping values and plots
[Jstop,para] = fctn(yStop);
Jstop = abs(Jstop) + (Jstop == 0);
Plots('stop',para);
end;

% evaluate objective function for starting values and plots
[Jc,para,dJ,H] = fctn(yc); H = H + 1 * eye(size(H));

Plots('start',para);
iter = 0; yOld = 0*yc; Jold = Jc; y0 = yc;

hisStr    = {'iter','J','Jold-J','|\nabla J|','|dy|','LS'};
his        = zeros(maxIter+2,6);
his(1,1:3) = [-1,Jstop,Jstop-Jc];
```

```
his(2,:)    = [0,Jc,Jstop-Jc,vecNorm(dJ),vecNorm(yc-yStop),0];

% some output
fprintf('%4s %-12s %-12s %-12s %-12s %4s\n%s\n',
    hisStr{:},char(ones(1,64)*'-'));
dispHis = @(var) fprintf('%4d %-12.4e %-12.3e %-12.3e %-12.3e
    %4d\n',var);
dispHis(his(1,:));

% -- end initialization

figure; spy(H);  % Représentation de la matrice

% Paramètres critère d'arrêt :
maxIterCG = 500; % Nombre maximum d'itérations GradConj et pcg
tolCG = 1e-6;    % Tolérance sur le résidu relatif
arret = 0;

% -- start the iteration
while arret == 0

    % check stopping rules
    STOP(1) = (iter>0) && abs(Jold-Jc)  <= tolJ*(1+abs(Jstop));
    STOP(2) = (iter>0) && (norm(yc-yOld)<= tolY*(1+norm(y0)));
    STOP(3) = (norm(dJ)                  <= tolG*(1+abs(Jstop)));
    STOP(4) = (norm(dJ)                  <= 1e6*eps);
    STOP(5) = (iter                     >= maxIter);

    if all(STOP(1:3)) || STOP(4) || STOP(5)
        arret = 1;
    end;

    iter = iter + 1;

    %---
    % solve the Gauss-Newton System
    if iter == 1
        % Résolution du 1er système par l'algo du gradient conjugué
        % Collecte de l'information pour construire les 3
            préconditionneurs LMP
        [dy,iteration,r,s,Hs,normr,Qout,Y,Z,theta,etaout] =
            GradConj(H,-dJ',maxIterCG,tolCG);

        % Construction du préconditionneur LMP
        switch LMP
              case 'QN'
```

```
132                     G = ConstrucQNFacto(s,Hs,r); % quasi-Newton-LMP
133                 case 'Ritz'
134                     G = ConstrucRitzFacto(theta,etaout,Qout,Z,Y); %
                            Ritz-LMP
135                 case 'Spectral'
136                     G = ConstrucSpecFacto(theta,Z); % spectral-LMP
137             end
138         else
139             z = @(r)MultiG(G,G',r);
140             [dy, flag, relres, iteration, normres] = pcg(H,-dJ',tolCG,
                    maxIterCG, z,[],yc);
141          end
142         %---
143         % check descent direction
144         % note: descent is not granted if using an iterative solver
145         descent =   dJ * dy;
146         if descent > 0,
147            warning('no descent direction, switch to -dy!')
148            dy      = -dy;
149         end;
150
151         % perform Armijo line-search
152         [t,yt,LSiter] =
               lineSearch(fctn,yc,dy,Jc,dJ,'LSMaxIter',LSMaxIter,...
153                        'LSreduction',LSreduction,'m',para.m,'omega',para.omega);
154         if (t == 0),
155             break;
156         end; % break if line-search fails
157
158         % save old values and update
159         yOld = yc; Jold = Jc; yc = yt;
160         [Jc,para,dJ,H] = fctn(yc); % evalute objective function
161         H = H + 1 * eye(size(H));
162
163         % some output
164         his(iter+2,:) =
               [iter,Jc,Jold-Jc,vecNorm(dJ),vecNorm(yc-yOld),LSiter];
165         dispHis(his(iter+1,:));
166         para.normdY = vecNorm(yc - yOld);
167
168         Plots(iter,para); % Animation recalage
169
170     end; % end of iteration loop
171
172     % ----------------------
173     Plots(iter,para);
```

```
174
175 % Critère d'arrêt:
176 His.str = hisStr;
177 His.his = his(1:iter+2,:);
178 fprintf('STOPPING:\n');
179 fprintf('%d[ %-10s=%16.8e <= %-25s=%16.8e]\n',STOP(1),'(Jold-Jc)',...
180         (Jold-Jc),'tolJ*(1+|Jstop|)',tolJ*(1+abs(Jstop)));
181 fprintf('%d[ %-10s=%16.8e <= %-25s=%16.8e]\n',STOP(2),'|yc-yOld|',...
182         norm(yc-yOld),'tolY*(1+norm(yc)) ',tolY*(1+norm(yc)));
183 fprintf('%d[ %-10s=%16.8e <= %-25s=%16.8e]\n',STOP(3),'|dJ|',...
184         norm(dJ),'tolG*(1+abs(Jstop))',tolG*(1+abs(Jstop)));
185 fprintf('%d[ %-10s=%16.8e <= %-25s=%16.8e]\n',STOP(4),...
186         'norm(dJ)',norm(dJ),'eps',1e3*eps);
187 fprintf('%d[ %-10s=  %-14d >= %-25s=  %-14d]\n',STOP(5),...
188         'iter',iter,'maxIter',maxIter);
189
190 FAIRmessage([mfilename,' : done !']);
191
192 end
193
194 %--------------------------------------------------------------------------
195
196 function [z] = MultiG(G,Gt,r)
197 z = G*Gt*r;
198 end
199
200 %--------------------------------------------------------------------------
```